## PROJECT PROMETHEUS by Lee Felsenstein

## HOW DOES YOUR CRYSTAL GROW by Sandra Faye

## MAKE ME MORE MUSIC, MAESTRO MICRO by Dorothy Siegel

# ROM

COMPUTER APPLICATIONS FOR LIVING

## THE WORDSLINGER
## 2200 Characters Per Second

Volume I, Number 5
November 1977/$2.00

# SWTPC announces first dual minifloppy kit under $1,000

SWTPC MINIFLOPPY DISK SYSTEM
SWTPC CT-64 TERMINAL SYSTEM
SWTPC 6800 COMPUTER SYSTEM

## Now SWTPC offers complete best-buy computer system with $995 dual minifloppy, $500 video terminal/monitor, $395 4K computer.

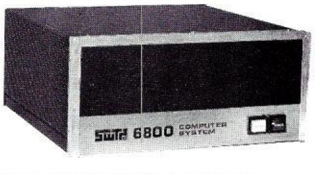
SWTPC MINIFLOPPY DISK SYSTEM

**$995 MF-68 Dual Minifloppy**
You need dual drives to get full benefits from a minifloppy. So we waited to offer a floppy until we could give you a dependable dual system at the right price.

The MF-68 is a complete top-quality minifloppy for your SWTPC Computer. The kit has controller, chassis, cover, power supply, cables, assembly instructions, two highly reliable Shugart drives, and a diskette with the Floppy Disk Operating System (FDOS) and disk BASIC. (A floppy is no better than its operating system, and the MF-68 has one of the best available.) An optional $850 MF-6X kit expands the system to four drives.

SWTPC CT-64 TERMINAL SYSTEM

**$500 Terminal/Monitor**
The CT-64 terminal kit offers these premium features: 64-character lines, upper/lower case letters, switchable control character printing, word highlighting, full cursor control, 110-1200 Baud serial interface, and many others. Separately the CT-64 is $325, the 12 MHz CT-VM monitor $175.



**$395 4K 6800 Computer**
The SWTPC 6800 comes complete with 4K memory, serial interface, power supply, chassis, famous Motorola MIKBUG® mini-operating system in read-only memory (ROM), and the most complete documentation with any computer kit. Our growing software library includes 4K and 8K BASIC (cassettes $4.95 and $9.95; paper tape $10.00 and $20.00). Extra memory, $100/4K or $250/8K.

**Other SWTPC peripherals** include $250 PR-40 Alphanumeric Line Printer (40 characters/line, 5 x 7 dot matrix, 75 line/minute speed, compatible with our 6800 computer and MITS/IMSAI); $79.50 AC-30 Cassette Interface System (writes/reads Kansas City standard tapes, controls two recorders, usable with other computers); and other peripherals now and to come.

**Enclosed is:**
_____ $1,990 for the full system shown above (MF-68 Minifloppy, CT-64 Terminal with CT-VM Monitor).
_____ $995 for the Dual Minifloppy
_____ $325 for the CT-64 Terminal
_____ $175 for the CT-VM Monitor
_____ $395 for the 4K 6800 Computer
_____ $250 for the PR-40 Line Printer
_____ $79.50 for AC-30 Cassette Interface
_____ Additional 4K memory boards at $100
_____ Additional 8K memory boards at $250
_____ Or BAC # __________ Exp. Date_____
_____ Or MC # __________ Exp. Date_____
Name__________ Address__________
City__________ State _____ Zip_____

SWTPC
Southwest Technical Products Corp.
219 W. Rhapsody, San Antonio, Texas 78216
**London:** Southwest Technical Products Co., Ltd.
**Tokyo:** Southwest Technical Products Corp./Japan

System 8813
PolyMorphic Systems
Off
On
Load
JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC

# The Computer for the Professional

Whether you are a manager, scientist, educator, lawyer, accountant or medical professional, the System 8813 will make you more productive in your profession. It can keep track of your receivables, project future sales, evaluate investment opportunities, or collect data in the laboratory.

Use the System 8813 to develop reports, analyze and store lists and schedules, or to teach others about computers. It is easily used by novices and experts alike.

Reliable hardware and sophisticated software make this system a useful tool. Several software packages are included with the machine: an advanced disk operating system supporting a powerful BASIC language interpreter, easy to use text editor, assembler and other system utilities. Prices for complete systems start at $3250.

See it at your local computer store or contact us at 460 Ward Dr., Santa Barbara, CA 93111, (805) 967-0468.

PolyMorphic
Systems

Sol Terminal Computer
Processor Technology
ESCAPE
TAB
BREAK
CTRL
SHIFT LOCK
SHIFT
UPPER CASE
LOCAL

# The Small Computer

Twenty-five years ago a computer as powerful as the new Processor Technology SOL-20/8 priced out at a cool million.

Now for only $1350 in kit form or $1850 fully assembled and tested you can have your own small computer with perhaps even more power. It comes in a package about the size of a typewriter. And there's nothing like it on the market today. Not from IBM, Burroughs, DEC, HP or anybody else!

## It fills a new role

If you're an engineer, scientist or businessman, the Sol-20 can help you solve many or all of your design problems, help you quantify research, and handle the books too. For not much more than the price of a good calculator, you can have high level computer power.

## Use it in the office, lab, plant or home

Sol-20 is a smart terminal for distributed processing. Sol-20 is a stand alone computer for data collection, handling and analysis. Sol-20 is a text editor. In fact, Sol-20 is the key element of a full fledged computer system including hardware, software and peripheral gear. It's a computer system with a keyboard, extra memory, I/O interfaces, factory backup, service notes, users group.

It's a computer you can take home after hours to play or create sophisticated games, do your personal books and taxes, and a whole host of other tasks.

Those of you who are familiar with small computers will recognize what an advance the Sol-20 is.

Sol-20 offers all these features as standard:

8080 microprocessor — 1024 character video display circuitry — control PROM memory — 9216 words of static low-power RAM — 2048 words of preprogrammed PROM — built-in cassette interface capable of controlling two recorders at 1200 bits per second — both parallel and serial standardized interface connectors — a complete power supply including ultra quiet fan — a beautiful case with solid walnut sides — software which includes a preprogrammed PROM personality module and a data cassette with BASIC-5 language plus two sophisticated computer video games — the ability to work with all S-100 bus products.

## Full expansion capability

Tailor the Sol-20 system to your applications with our complete line of peripheral products. These include the video monitor, audio cassette and digital tape systems, dual floppy disc system, expansion memories, and interfaces.

## Write for our new 22 page catalog. Get all the details.

Processor Technology, Box O, 6200 Hollis St.,
Emeryville, CA 94608. (415) 652-8080.

**Processor Technology**

Volume I, Number 5

# Contents

November 1977

## FEATURES

**22 Project Prometheus: Going Solar with Your Micro** **by Lee Felsenstein**
A complete guide and construction tutorial on using your micro for solar energy measurement. Plus great hopes for the future.

**36 BASIC from the Word GOTO** **by Eben Ostby**
A beginner's introduction to the current Everyman's microcomputer language.

**42 Chipmaker, Chipmaker, How Does Your Crystal Grow?** **by Sandra Faye**
An inside look at growing silicon crystals, the heart of today's intelligent chip.

**48 Robotic Verse** **by Gloria Maxson**
Some not so mechanical limmericks to lighten your day.

**54 The Kit and I, Part Three: Personality Plus** **by Richard W. Langer**
The continuing saga of one man's struggle with soldering iron and components.

**58 Make Me More Music, Maestro Micro** **by Dorothy Siegel**
More music to play on your computer. Plus a duet so you can join in.

**64 Wings in Wind Tunnels: Computer Models and Theories** **by Joseph Weizenbaum**
Theories need models and models generate theories. Together they give intelligent guidance for solving problems.

**75 What is a Microcomputer System?** **by Leslie Solomon and Stanley Veit**
A basic introduction to putting together a personal computer system.

**82 Maintaining Your Micro** **by O. S. (The Old Soldier)**
That little matter of preventive maintenance and troubleshooting.

**86 Time Sharing on the Family Micro** **by Barry Yarkon**
Two (or more) can play as cheaply as one.

**90 Conversion of the Diggers** **by Dimitri V. Gat**
A short story.

89-92 missing

**94 The Wordslinger: 2200 Characters Per Second** **by Stuart Dambrot**
The fastest printer in town.

**96 Light Fantastic: The Kinetic Sculpture of Michael Mayock** **by Tom Moldvay and Lawrence Schick**
A self-taught computerist breaks the rules. The result? Flashing lights and electronic art.

## COLUMNS

**10 Missionary Position by Theodor Nelson**
Kids, Parents, and Computers

**12 The Human Factor by Andrew Singer**
Tiny FORTRAN IV

**16 PROMqueries by Eben Ostby**
Have a Problem? Ask ROM

**17 Legal ROMifications by Peter Feilbogen**
Getting around the Post Office

**19 AIQuotient by Stuart Dambrot**
Perspective on Perception

**99 FutuROMa by Bill Etra**
The Eye of the Daphnia

## DEPARTMENTS

**4 On the Bus**

**8 Reader Interrupt**

9-13 missing

**14 Babbage and Lovelace**

**17 Eve 'n' Parity**

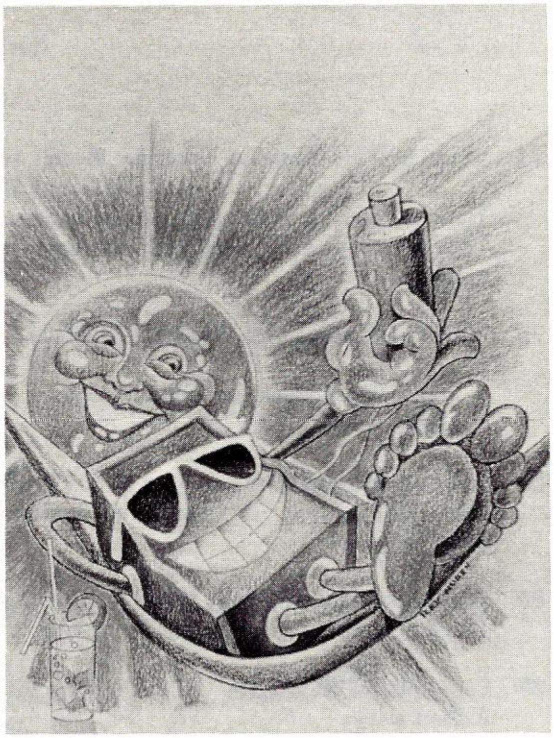
**49 Run On Micros**

**50 Centerfold**

**100 PROMpuzzle**

*ROM* is published monthly by ROM Publications Corporation, Route 97, Hampton, CT 06247 (Tel. 203-455-9591). Domestic subscriptions are $15 for one year, $28 for two years, and $39 for three years. Canada and Mexico $17 for one year, $30 for two years, and $41 for three years. For European and South American subscriptions, please add $12 per year additional postage. For all other continents, please add $24 per year additional postage. *Copyright © 1977 by ROM Publications Corporation. All rights reserved.* Reproduction in any form or by any means of any portion of this periodical without the written consent of the publisher is strictly prohibited. The following trademarks are pending: AIquotient, Babbage and Lovelace, Eve 'n' Parity, floppyROM, futuROMa, The Human Factor, Legal ROMifications, Missionary Position, The Noisy Channel, On the Bus, PROMpuzzle, PROMqueries, Reader Interrupt, ROMdisk, ROMshelf, ROMtutorial, and Run On Micros. Opinions expressed by authors are not necessarily those of *ROM* magazine, its editors, staff, or employees. No warranties or guarantees explicit or implied are intended by publication. Application to mail at second-class postage rate pending at Hampton, CT 06247. Membership in Audit Bureau of Circulation pending.

# On the Bus



**Rex Ruden** started out by drawing cowboys, but he didn't know how to do the feet, so he had to put his favorite guntoters behind rocks or knee-deep in grass in all his early efforts. Since elementary school days, however, feet have ceased to be a problem. His experience has been mostly in display, advertising, and animation. He also worked with well-known black animator Tee Collins. After spending the last five years with Aetna in the company's audio-visual department, Ruden plans to work more and more on magazines and children's illustration.

Hooked on crossword puzzles at an early age, **Daniel Alber** now constructs as well as solves them. Part of the Brownstone Renovation generation in New York, when he's not constructing puzzles for the likes of *Field and Stream, The New York Times,* and *ROM,* he's reconstructing olden golden rooms in his Brooklyn based house.

**Stuart Dambrot** has had a wide-ranging academic career. He is currently at the University of Connecticut. Disillusioned with traditional psychological theories, he has become interested in computer science and artificial intelligence. Stuart is now trying to develop alternative hypotheses about the nature of human cognition.

**Bill Etra** is a West Coast based computer design consultant. He is co-inventor of the Rutt/Etra Video Synthesizer—the first portable voltage control analog video synthesizer, as well as the Videolab. His main interest is videographics, and many of his works have appeared as cover illustrations on various periodicals and books including *Computers in Society* and *Broadcast Management and Engineering*. His current research centers on "The Computer as a Compositional Tool for Video."

**Sandra Faye** is currently a contributing editor and The Underground Gourmet for *New West* in San Francisco. She has written for such diverse publications as *The New York Times Book Review, City, Earth Times,* and *Oui* on subjects ranging from carcinogens and energy issues to ethnic cuisines.

**Peter Feilbogen** attended the Rutgers School of Business Administration and Brooklyn Law School. In addition to being an attorney, he is also a Certified Public Accountant. He has been a sole practitioner on Long Island for approximately ten years, and treasurer of Data Information Services, Inc. An avid tennis player, he is currently trying to improve his game with the aid of a computer.

**Lee Felsenstein** was born in Philadelphia and grew up wanting to be an inventor. Outside of that, he bears no resemblance to W. C. Fields whatsoever. Instrumental in establishing the first experimental public-access information-exchange system in 1972, he is presently engaged in further development in that area of communications. In his spare time he has designed the Pennywhistle 103 modem, the VDM-1 video display module, the Sol terminal/computer, and the Vid-80 video display card. Lee was also instrumental in forming the original Homebrew Computer Club and currently serves as its "toastmaster."

A member of the Science Fiction Writers of America, **Dimitri V. Gat** has taught both technical and creative writing. His publications include a science fiction novel and other fiction and nonfiction, including a book on baseball. He became interested in computers while doing technical writing and he considers himself a generalist with scientific interests.

**Richard W. Langer** is a free lance writer whose articles have appeared in such diverse magazines as *New York, Family Circle, House and Garden,* and *Esquire*. Currently he is a columnist with *The New York Times* and at work on his fifth book.

**Gloria Maxson's** fascination with robots began when, as a college freshman at Santa Barbara, she read Chekhov's *R. U. R.,* a tragic play about automatons. Her robots, "Glorobots," however, are a much more likeable race

**Tom Moldvay** is an aspiring free lance writer in Kent, Ohio, where he went in 1972 to do graduate work in anthropology and to teach in the experimental college at Kent State. He was co-editor of the Kent science fiction and fantasy magazine, *Infinite Dreams,* where he has published stories.

**Theodor Nelson** is the author of the classic *Computer Lib/Dream Machines,* a Whole Earth style catalogue of computer machinations. Presently at the Department of Mathematics of Swarthmore College where he is working on the Hypertext Project, Ted specializes in highly interactive systems for graphics and text. His past experience includes a stint at Dr. Lilly's Dolphin Laboratory and work as a consultant for Bell Lab's ABM system.

**The Old Soldier**, who actually has military background, has written about computers since the early days of the micro (1974). His articles of opinion and advice have appeared in the pages of *People's Computer Company* (now *People's Computers*). A mechanical engineer by profession, he is noted at the Homebrew Club for his Sphere personal computer, which he uses for business and recreation.

**Eben Ostby** first began experimenting with computers while at the Pomfret School, which had a PDP-8. He went on to become the first Computer Science major at Vassar College, from which he recently graduated with honors. Ostby has done extensive work with graphics in APL, and recently programmed what he thinks may be the first cartographic project in APL.

A science fiction and fantasy book seller who leans toward free-lance writing, **Lawrence Schick** is currently attending Kent State. His achievements include co-editing the Kent State amateur science fiction magazine, being a plaintiff in an ACLU test case (he lost), and once running a science fiction convention.

**Dorothy Siegel**, a graduate of Smith College and Yale University, is interested in both classical music and computer applications. Her collection of musical instruments includes, beside her clarinet, an IMSAI computer with floppy disk, Diablo printer, and Model 6 Music Board—the last manufactured by her company, Newtech Computer Systems, Inc., of Brooklyn, New York.

**Andrew Singer** has been hooked on computers since he first built one in 1958. A hard/software consultant, he is fluent in thirty computer languages and knows more than enough about twenty species of machine. His work has included the first medical information retrieval system based on ordinary clinical records, and a large and intricate system for interactive selection of data from public opinion polls. He believes that most software is poorly designed and unspeakably rude, and his Ph.D. research is aimed at improving the architecture and human engineering of interactive systems.

**Leslie Solomon** is Technical Editor of *Popular Electronics,* where he devotes half his time to computer developments. He introduced the first personal computer, the Altair, and later many others, including, most recently, Speechlab, the first vocal interface for a personal computer.

**Stanley Veit** has been in the computer field for about twenty years, originally as a technical writer and instructor of computer maintenance. He is coauthor (with Leslie Solomon, Technical Editor of *Popular Electronics*) of *Getting Involved With Your Own Computer.* Veit founded the first computer store on the East Coast.

**Joseph Weizenbaum** is Professor of Computer Science at the Massachusetts Institute of Technology. He is best known to his colleagues as the composer of SLIP, a list-processing computer language, and for ELIZA, a natural-language processing system. More recently, he has directed his attention to the impact of science and technology—and of the computer in particular—on society.

**Barry Yarkon** is vice president of a New York based phototypesetting firm. He is a frequent contributor to trade publications on computer typesetting technology and teaches several courses and seminars around the country. Formerly a medical editor for Harper & Row, Inc., Barry (a graduate pharmacist) is self-taught in computer science. In his "spare time," he is researching mini-database management techniques and languages for "small files on small computers." ▼

---

*A Note to Authors:*
ROM is always looking for good computer applications articles from people with up-and-running systems. We also will be glad to consider for possible publication manuscripts, drawings, and photographs on other computer-related subjects. Manuscripts should be typewritten double spaced, and a stamped self-addressed envelope of the appropriate size should accompany each unsolicited submission. Although we cannot assume responsibility for loss or damage, all material will be treated with care while in our hands. Manuscripts should be sent to ROM Publications Corporation, Route 97, Hampton, CT 06247.

RAINBOW COMPUTING, INC.

Supplier of

WAVE MATE
THE DIGITAL GROUP
DEC PDP
Computer products

Peripherals and Supplies from

PERSCI
CENTRONIX
DIABLO
MAXELL
COMPUTER DEVICES
LEAR-SIEGLER
MULTI-TECH
TEXAS INSTRUMENTS

Specialists in Design, Implementation
and Support of
Custom Hardware/Software Systems for
Business, Educational, and Personal Use

Experts in most major computer
software including
CDC, IBM, PDP
BASIC, COBOL, FORTRAN, PL1
LISP, SIMULA, SNOBOL, SPSS, BMD's
COMPASS, MACRO,
6800, & Z80 assembly languages

10723 White Oak Ave., Granada Hills,
Ca. 91344
(213) 360-2171

**Reconditioned Teletypes**
**Guaranteed 90-days on return basis**
**ASCR 33 $950 KSR 33 $595**

**Free delivery within 25 mile**
**radius of New York City.**
**Shipping extra beyond radius.**

**IMSAI edge connector and guides:**
**ten for $39, $4.50 each.**

**COMPUTER MART OF**
**NEW YORK**
**118 MADISON AVE**
**NEW YORK, NEW YORK 10016**

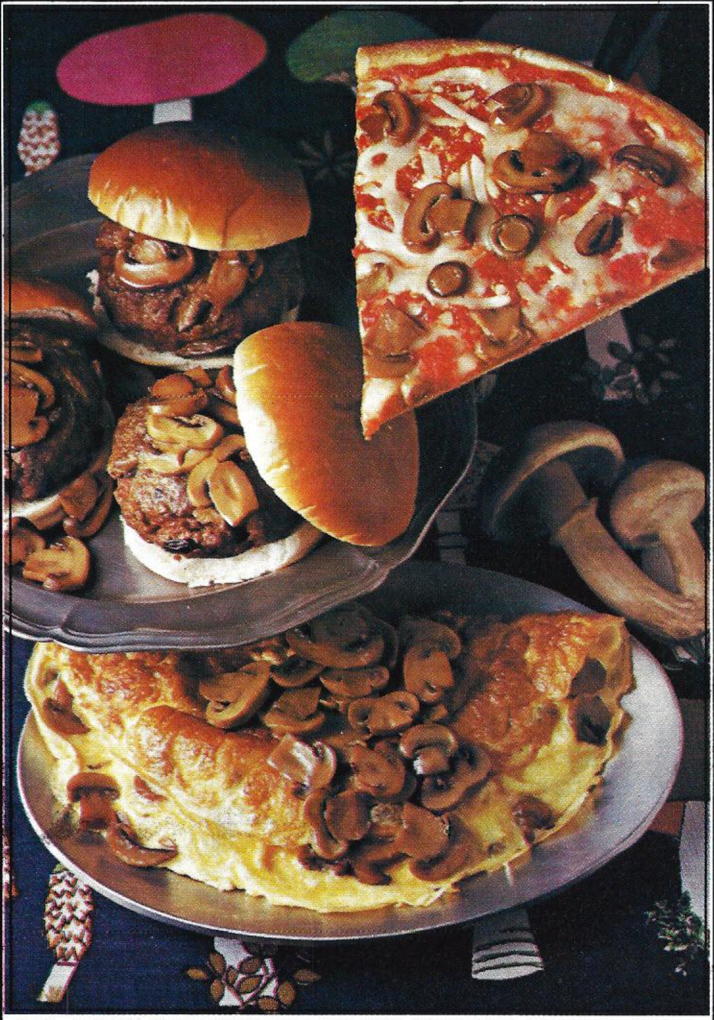
**(212) 686-7923**

# COLLIER IS THE BEST ENGRAVER, PRINTER IN THE COUNTRY.

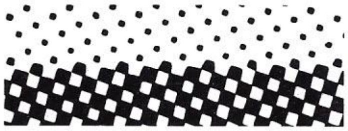
Because. A revolution in engraving has happened. Collier split the dot. And because of the new Laser Beam, Collier is now light years ahead of the rest. The laser does it. Here's how it works. Technically, the laser beam which is used to control film exposure (thus "Program" the engraving) is split into six sectional beams. Each of these is digitally modulated by computer to transfer half-dots of picture information per scanner revolution. Since two picture-information "bits" are available per dot in circumferential direction, it's possible to expose a smaller area in the second "orbit"...or even completely omit it (thus producing an actual half-dot or even an elliptical mini-dot). If that seems to all sound a lot like gobbledygook, don't worry about it. The important thing is, it's here and it does work and it gives your image resolution that you never could get before. We'll be happy to demonstrate it for you. Even happier to produce your next set of four-color letterpress, offset and gravure engravings. Call Collier Graphic Service Company Inc., 240 West 40th Street, New York, New York 10018/(212) 840-0440.

| ATLANTA | BOSTON | DETROIT | HARTFORD | NEW YORK | PHILADELPHIA |
|---|---|---|---|---|---|
| (404) 892-2383 | (617) 965-5660 | (313) 259-2111 | (203) 367-0706 | (212) 840-0440 | (215) 988-0110 |

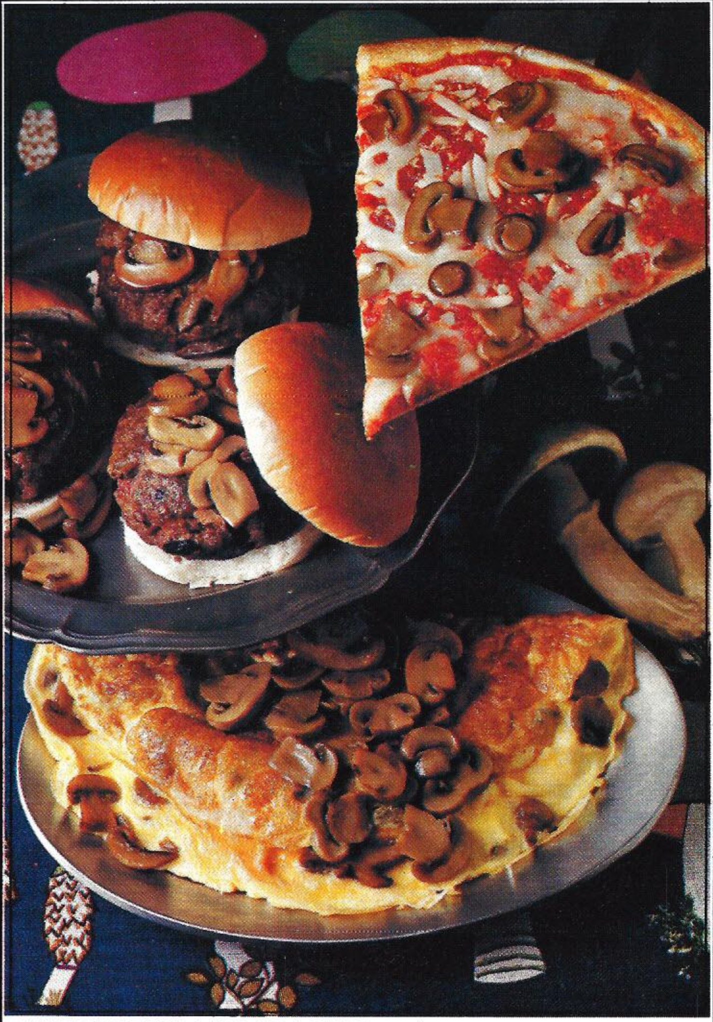
OR CALL TOLL FREE (800) 221-2585/(800) 221-2586



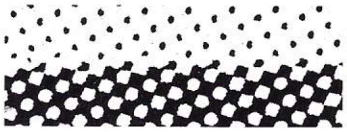
THE ABOVE ENGRAVING WAS MADE WITH THE CONVENTIONAL PLATE MAKING PROCEDURES, AS YOU CAN SEE, IT LACKS COLOR FIDELITY AND SHARPNESS, IF YOU COMPARE IT WITH…



THE CONVENTIONAL ENGRAVING DOT.



THE COLLIER'S LASER BEAM ENGRAVING, AS YOU CAN SEE FROM THE ABOVE, HAS BRILLIANCE, SHARPNESS AND COLOR FIDELITY THAT ONLY COLLIER'S DOT CAN PROVIDE.



THE COLLIER LASER DOT.

THE COLLIER ADVANTAGE. COLLIER GRAPHIC SERVICES COMPANY, INC., 240 WEST 40TH STREET, NEW YORK, NEW YORK 10018

# Reader Interrupt

Dear ROM,

In regard to Richard Langer's difficulties with clipping off component leads ("The Kit & I," August '77); the trick is to fill the space between the cutting edges of the wire cutters with Dow Corning Silastic, let it cure a couple of days, and then split the Silastic with a razor blade. The Silastic will catch the cut wires without hampering the operation of the cutters.

Austin Massey
Colorado Springs, Colorado

P. S. Sorry I can't give credit to whoever thought of this idea. It's been around so long I forgot. Also there are cutters on the market with rubbery goo already installed by the manufacturer.

Dear ROM,

A good friend of mine, Ted Lewis, recently gave me his copy of *ROM.* It was thoroughly enjoyable reading it. A delightful change from all the other "What I learned from building kit x and y" and "Here is code to play game abc on machine i in mini-tiny-extra-small-BASIC" type magazines. Your articles were thought-provoking and stimulating!

I do have two suggestions, however. First, the name *ROM* is lousy. It should have been *RWM*—for at least two reasons: (1) you want your readers to *Write* to you, the authors, and the advertisers. (2) As I said before, the articles are thought-provoking and the reader often wants to write down a comment like "Good," "Blah," or "that makes me think of. . . ." In fact I suggest you change your format by leaving at least one and one-half inch margins at the top and side of all articles to allow (and encourage) the reader to participate by evaluating and writing. Secondly, I think you should have each author, in the spirit of *Scientific American,* prepare a small, but relevant, and perhaps annotated, bibliography on their article—to guide the reader into a particular topic.

**DON'T FORGET THE DIGITAL FOAM CONTEST**

Beside our regular payment to contributors, we will be giving away two prizes for the best applications article on digital foam we receive before January 1, 1978.

First prize is your choice of $500 or its equivalent in Dynacon. Second prize is $250 or its equivalent in Dynacon.

The articles should be 1,500-3,000 words long. They should include diagrams (roughs are fine) and photographs of your system in operation. You must be able to demonstrate that you have it up and running.

For further details see *ROM,* Vol.I, No.1.

Dynacon material kits for experimentation and implementation are available from:

Dynacon Industries Inc.
14 Bisset Drive
West Milford, N.J. 07480

for $10 if you send a check with your order, or $12 if you wish to be billed.

Now, having read my suggestions, please enter my subscription for 1 year. Send it to:

Bruce D. Shriver
Layfayette, Louisiana

Dear ROM,

The concept of binary clocks described in your second issue is the most amusing idea I've come across in ages. Would that we could install one to replace Big Ben. The problem, of course, is the chimes. How would we get a proper binary bong?

Chester Thompson
London, England

Dear ROM,

I noticed a slight error in the new August issue; the hungry gentleman on the cover was inserting the delicious floppy disk into his mouth in the wrong direction. To obtain the greatest satisfaction from a disk, it must be inserted correctly. Obviously the artist is not a gourmet such as myself!

W. Bradfield Higgins
New York, New York

*You know the old story. If you don't feed your computer right, it's just going to gobble up those disks.*
*ROMulus*

ROM
COMPUTER APPLICATIONS FOR LIVING

**Editor and Publisher**
Erik Sandberg-Diment

**West Coast Editor**
Lee Felsenstein

**Associate Editors**
Sue Neillson
Janet C. Robertson

**Assistant Editor**
Karen K. Jambeck

**Contributing Editors**
Sandra Faye
Bill Etra
Louise Etra
Ed Hershberger
Avery Johnson
Richard W. Langer
Theodor Nelson
Robert Osband
Eben Ostby
Frederik Pohl
Andrew Singer
Alvin Toffler

**Crossword Puzzle Editor**
Daniel Alber

**Editorial Assistant**
Donna Parson

**Art Director**
Susan Reid

**Contributing Artists**
Steve Gerling
Robert Grossman
Cindy Hain
Luis Jimenez
Korkie
Rex Ruden
Linda Smythe

**Art Assistant**
Sue Bass

**Staff Photographer**
Thomas Hall

**Composition**
Lynn A. Archer

**Special Assistants**
Jennifer L. Burr
Joanne Zeiger

**Counsel**
Peter Feilbogen

# PCE PERSONAL COMPUTING EXPO
## NEW YORK COLISEUM, OCTOBER 28, 29, 30, 1977

## General Information

You may find it advantageous to purchase two or three-day admission tickets in advance. These are available by mail only, no later than October 10, 1977. Use coupon below.

Group rates (10 or more persons) qualify for $1.00 off regular prices. Arrangements must be made by mail prior to October 10, 1977.

Special arrangements have been made if you desire to stay overnight. Our headquarters hotel, the Barbizon-Plaza, is located on Central Park South, two blocks from Columbus Circle. Single rooms available at $34.00 per night; $40.00 double, plus tax. There's a weekend plan: $22.95 daily, plus tax per person, double occupancy . . . includes breakfast (brunch on Sunday) and meal gratuities. Children under 14 in same room with parents, free.

For hotel reservations and information, call toll free (800) 223-5493. From New York State call (800) 223-5963.

For those traveling to New York by air, American Airlines offers a convenient service through arrangement with Personal Computing Expo. For information, call toll free (800) 433-1790. In Texas the number is (800) 792-1150. From the West Coast, round trip fare via American is only $227.00.

20,000 persons are expected to attend and view the more than 200 exhibits by personal computer manufacturers and retailers.

Personal Computing Expo will occupy the 4th floor of the New York Coliseum. It is located on 59th Street and Columbus Circle — the geographical center of New York City. Garage parking in the building is available.

For answers to any questions pertaining to your attendance at Personal Computing Expo, contact the Show Manager, Ralph Ianuzzi, at Area Code 212/753-4920.

## Show Hours and Admission

Personal Computing Expo hours are as follows:
Friday, Oct. 28 — Noon to 10 p.m.
Sat. Oct. 29 — 10 a.m. to 10 p.m.
Sunday, Oct. 30 — Noon to 7 p.m.
General Admission: $5.00 (includes free BYTE lectures) per day.
Two-day Tickets: $9.00 (advance sale only)
Three-day tickets: $13.00 (advance sale only)

## Exciting lectures sponsored by BYTE

Personal Computing Expo is endorsed by BYTE magazine, whose staff has contacted prominent speakers for an exciting series of lectures.
Visitors will be able to attend these meetings **free of charge.** The lectures will not conflict with each other eliminating the worrysome choice among several equally important topics. In addition, they will be repeated on the next day to give you a second chance if you missed a topic.
Lectures are typically 30 minutes, often with demonstrations and an additional 15 minutes for questions.

**Louis E. Frenzel**

**Jack L. Davies**

**Carl Helmers** **John H. Dilks III** **David Fylstra** **Carl L. Holder**

**Sol Libes** **Portia Isaacson Ph.D.** **Max Mathews Ph.D.** **Robert S. Jones**

---

Personal Computing Expo admission is $5.00 per day. Advance reservation eliminates waiting in line. Order advance tickets with this coupon. Admission ticket includes access to exhibits, lectures and tutorials.

Please send me ________________ advance registration tickets for three days, October 28-29-30. Total cost $13.00 per person.
Please send me ________________ advance tickets for two days, October ______ and October ______. Cost is $9.00 per person.
Please send me ________________ advance tickets for one day, October ______. Cost is $5.00 per person.

**Make all checks payable to PERSONAL COMPUTING EXPO, and mail to:**
**Personal Computing Expo, 78 East 56th Street, New York, N.Y. 10022.**

Name ______________________________ Amount enclosed $____________
Address ______________________________________________
City ____________________ State ______________ Zip ____________

# PROMqueries

# HAVE A PROBLEM? ASK ROM

### by Eben Ostby

Dear ROM:

A friend of mine, who lives 100 miles away in the next state, has a personal computer, as do I. We would like to interchange programs and data and have the computers interact somehow. Is there an easy way to hook up computers over phone lines?

James McLaughlin
Denver, Colorado

*Dear James,*

*There are dozens of ways to interface computers over phone lines, and some of them are easy and cheap. The easiest, however, are probably not the cheapest. The most flexible method would be to use a* modem *(MOdulator-DEModulator) at each end of the phone line. One end would need the type of modem that answers calls on a time-sharing system; it provides the carrier tone when you connect with it. The other end could use a standard acoustic coupler attached to the RS-232 interface on your computer. This system has the advantage of being standardized and reliable. It is also potentially very expensive. We've seen one modem that would be perfect for this application. The D C Hayes 80-103A Data Communications Adapter, which is ready to plug into your S-100 bus, costs about $280 and is available from D. C. Hayes, Inc., P. O. Box 9884, Atlanta, GA 30319. An old acoustic coupler can probably be picked up for a song or two from your local warehouse sales outlet or computer store.*

*A cheaper though less certain way of communicating by phone involves a modified cassette interface. It may take a lot of experimentation to get it to work (we've never tried), but it should be possible to hook up a Kansas City-type interface to a small amplifier and speaker, and hold the speaker to the phone. At the other end, you'd need a similar interface on your friend's computer. What you'd be doing—if it worked—would be fooling one computer into thinking that it was writing on tape, while it was actually speaking into the telephone. The other end would be fooled into thinking that it was reading a tape. You, the users, would have to be very careful with your protocol. What I mean by this is that you'd have to synchronize your movements carefully, or you might find that your data was not being received. You'll probably have to make a great many trials before you get the bugs off the phone.*

*By the way, if you get it to work, let us know.*

ROM

Dear ROM,

I'm confused about all the different new processing techniques heralded lately in journals such as yours. What are the differences between parallel processing, serial processing, and matrix processing?

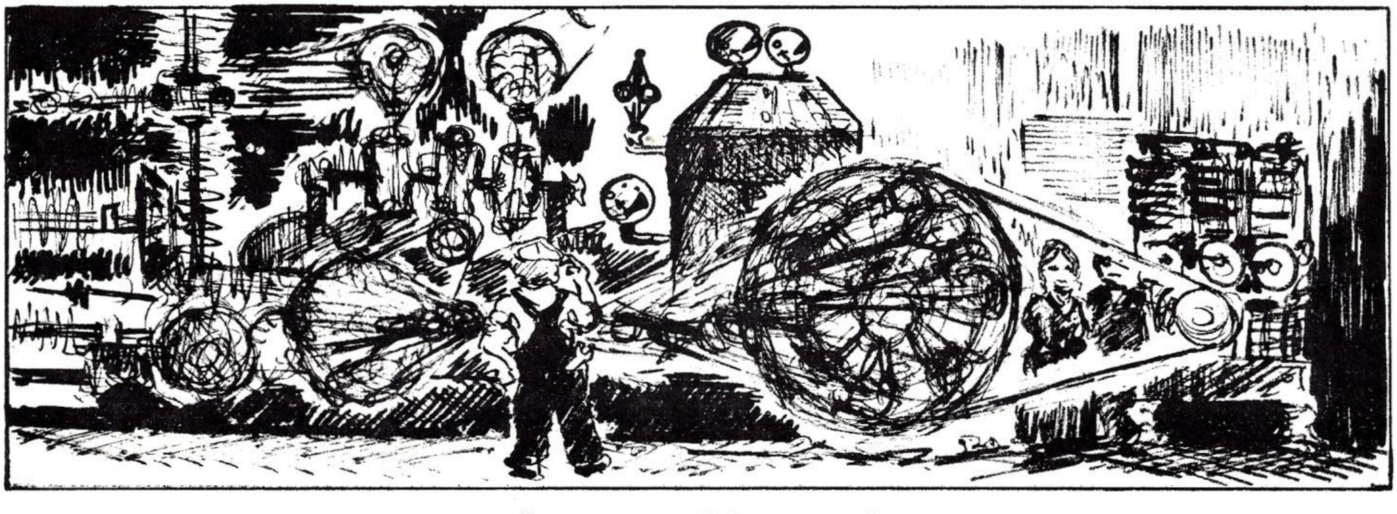
Robert Swanson
Minneapolis, Minnesota

*Dear Robert,*

*If you're referring to the way different languages handle data, the differences are really illusory. Even though some languages appear to process arrays in parallel—meaning they appear to operate on all elements of the array simultaneously—most computers actually operate on one element at a time. The high-level software allows you to tell the computer to add the whole array to another one, but the computer ends up doing a separate addition for each element. There is a savings in your own time, though, since you only have to specify one array addition.*

---

**BABBAGE AND LOVELACE**



*"Watt says it will do 500 rpm."*

Cartoon by Steve Gerling

*Matrix processing is another extension of parallel processing in this sense. Matrix processing is simply the processing of a two-dimensional array in one blow.*

*There are other meanings for the expressions* parallel processing *and* serial processing. *They sometimes refer to the way in which a computer operates on its own data inside the processor (CPU). On a serial machine, additions, comparisons, ANDs, and all other operations take place serially* bit by bit. *Since the hardware designer need only build a one-bit adder instead of an eight-, twelve-, or sixteen-bit adder, the cost of a serial machine is likely to be much less than that of a parallel machine, where all eight bits of a computer word or byte are added simultaneously. A serial machine is also much slower than a parallel machine—imagine eight people each adding one pair of numbers versus one person adding eight pairs of numbers.*

*Parallel processing has reached its apex in the experimental Star computer, which does parallel array operations with hardware instead of software. The Star, in effect, adds many pairs of bytes all at one time.*

ROM

Dear ROM,

So many systems are described as coming with "all the bells and whistles" attached. Where did this expression originate?

Jonathan Strick
Philadelphia, Pennsylvania

*Dear Jonathan,*

*I haven't the vaguest idea.*

ROM

Dear ROM,

I have a research library of approximately 2,000 volumes with roughly 60,000 items I'd like referenced and cross-referenced. Hopefully this would save me more time in the long run than simply entering all the data would. Also I add perhaps 150 journals a year to my collection, each with an average of ten relevant articles.

How big an undertaking would it be, first of all, to develop a program for this? Secondly, from what I've seen, data on floppy discs seems to vanish at the most inopportune times. How do I protect my data once I've got it? And, lastly, is all this even a feasible idea for a microcomputer system?

Ian White
Berkeley, California

*Dear Ian,*

*This sounds like an ideal project for someone with lots of space on a floppy. Actually, two floppies would be more like it. If you figure, say, thirty characters per title, twenty characters per author's name, fifteen characters for a reference number, and another ten for subject category and other flags, each book will require some seventy-five bytes of information—perhaps less if you pack it in tightly. At that rate, you would have 150K bytes of data for your library—much less than a typical full-size floppy holds. But, as you suggest, data stored on a floppy may be less than archival. My suggestion is that you resort to the primitive technique of "backing up" your data, that is, make a copy of your precious files and store it away along with copies of the transactions you entered to modify that copy of the file. This way, if your system bombs and you find out about it (which is no sure thing), you can re-enter the data you entered after you "backed up" the system. In fact, this is a good idea no matter what data you're storing on your disks.*

*I can't, unfortunately, estimate how large a program would be needed to automate your collection. Many factors determine the size of the program for any given task.*

*The number of features your program will encompass makes the biggest difference. You could write a tiny program to read and write records with your book information on them. But you probably want to have sorted lists and searches for books by author, title, subject, and (if you lend books) borrower. As you add this information, both the program and the data space grow proportionately.*

*Facilities for user input tend to be costly in terms of programming complexity. If you restrict the user to typing in a few cryptic symbols for access to data, your input routines, including the ever-present error checking routines, are likely to be fairly small and simple. But if you'd like to be able to ask for a book in simple English, or specify all your options at once, or some such thing, you would have to*

*"But, Babbage dear, what about the glitches?"*

*"Second-generation logic."*

*write a monstrous input routine just to handle possible errors.*

*Packing data often takes a lot of programming effort. "You don't get something for nothing." It is a hard and fast rule that the more compact you want your data to be, the larger and more complex the programs that operate on that data will have to be. All the encoding, packing, decoding, and unpacking routines take up space—and time.*

*Depending, then, on what sort of system you want, and what you have available in the way of memory and peripherals, you should be able to create a very respectable library system. Maybe you could sell it to others like yourself.*

ROM

Dear ROM,

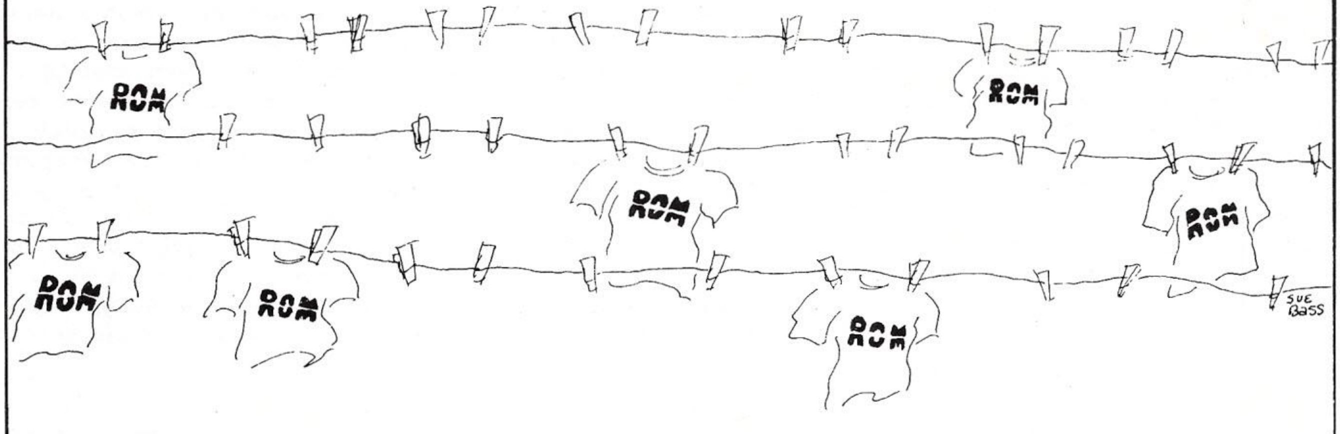
Where I live, we have frequent power failures—not of the New York City variety, but limited, short-term brownouts. Is there a way I can protect my micro in the event of such a failure?

Powerless
Storrs, Connecticut

*Dear Powerless,*

*If you only wish to protect yourself against short-term voltage drops—say down to 90 v—you can buy a voltage regulator, a box which plugs in between your computer and the wall. It won't help if the power goes off for a second, though—then you'll still lose your data. If this is likely in your area, you might try running the computer on batteries. If that isn't feasible, look for a computer attachment that will cause an interrupt when power levels drop. This attachment monitors the line voltage, and when it drops below a certain level, the device issues an I/O interrupt to the processor. Such devices were common on minis some years back although lately better back-up power systems have made them less necessary. If you were to design a low-power interrupt system for, say, an 8080-based machine, the power monitor would trigger the interrupt in the computer when the power dropped below the level necessary to maintain proper system performance. The monitor would also issue a single computer instruction to the processor, which would execute that instruction immediately. Usually, a subroutine jump instruction (RTS on the 8080) transfers program execution to a routine that quickly puts the system into a state where only a minimum amount of data can be lost. If time permits—you would probably have a few milliseconds before the power was too low for operation—your routine can try to dump important parts of memory to a floppy. The whole idea is not to avoid the unavoidable but to make restoring the system as easy as possible when the lights come back on.*

ROM



**Get your ROM T-shirt now!**

**Order from:**
**ROM Publications Corp.**
**Route 97**
**Hampton, CT 06247**

**Please ship to:**
**Name** ______
**Address** ______
**City** ______ **State** ______ **Zip** ______

☐**S** **Qty.** ______
☐**M** **Qty.** ______
☐**L** **Qty.** ______ ☐**Child S** **Qty.** ______
☐**XL** **Qty.** ______ ☐**Child L** **Qty.** ______
☐**Check or money order enclosed.**
☐**Master Charge** ☐**BankAmericard**
**Exp. date** **Card#**

**$5 ppd.**
**2 for $9 ppd.**

**Please allow 4 to 6 weeks for delivery.**

# PCE PERSONAL COMPUTING EXPO
## NEW YORK COLISEUM, OCTOBER 28, 29, 30, 1977

## General Information

You may find it advantageous to purchase two or three-day admission tickets in advance. These are available by mail only, no later than October 10, 1977. Use coupon below.

Group rates (10 or more persons) qualify for $1.00 off regular prices. Arrangements must be made by mail prior to October 10, 1977.

Special arrangements have been made if you desire to stay overnight. Our headquarters hotel, the Barbizon-Plaza, is located on Central Park South, two blocks from Columbus Circle. Single rooms available at $34.00 per night; $40.00 double, plus tax. There's a weekend plan: $22.95 daily, plus tax per person, double occupancy . . . includes breakfast (brunch on Sunday) and meal gratuities. Children under 14 in same room with parents, free.

For hotel reservations and information, call toll free (800) 223-5493. From New York State call (800) 223-5963.

For those traveling to New York by air, American Airlines offers a convenient service through arrangement with Personal Computing Expo. For information, call toll free (800) 433-1790. In Texas the number is (800) 792-1150. From the West Coast, round trip fare via American is only $227.00.

20,000 persons are expected to attend and view the more than 200 exhibits by personal computer manufacturers and retailers.

Personal Computing Expo will occupy the 4th floor of the New York Coliseum. It is located on 59th Street and Columbus Circle — the geographical center of New York City. Garage parking in the building is available.

For answers to any questions pertaining to your attendance at Personal Computing Expo, contact the Show Manager, Ralph Ianuzzi, at Area Code 212/753-4920.

## Show Hours and Admission

Personal Computing Expo hours are as follows:
Friday, Oct. 28 — Noon to 10 p.m.
Sat. Oct. 29 — 10 a.m. to 10 p.m.
Sunday, Oct. 30 — Noon to 7 p.m.
General Admission: $5.00 (includes free BYTE lectures) per day.
Two-day Tickets: $9.00 (advance sale only)
Three-day tickets: $13.00 (advance sale only)

## Exciting lectures sponsored by BYTE

Personal Computing Expo is endorsed by BYTE magazine, whose staff has contacted prominent speakers for an exciting series of lectures.
Visitors will be able to attend these meetings **free of charge.** The lectures will not conflict with each other eliminating the worrysome choice among several equally important topics. In addition, they will be repeated on the next day to give you a second chance if you missed a topic.
Lectures are typically 30 minutes, often with demonstrations and an additional 15 minutes for questions.

**Louis E. Frenzel**

**Jack L. Davies**

**Carl Helmers** **John H. Dilks III** **David Fylstra** **Carl L. Holder**

**Sol Libes** **Portia Isaacson Ph.D.** **Max Mathews Ph.D.** **Robert S. Jones**

Personal Computing Expo admission is $5.00 per day. Advance reservation eliminates waiting in line. Order advance tickets with this coupon. Admission ticket includes access to exhibits, lectures and tutorials.

Please send me ________________ advance registration tickets for three days, October 28-29-30. Total cost $13.00 per person.
Please send me ________________ advance tickets for two days, October ______ and October ______. Cost is $9.00 per person.
Please send me ________________ advance tickets for one day, October ______. Cost is $5.00 per person.

**Make all checks payable to PERSONAL COMPUTING EXPO, and mail to:**
**Personal Computing Expo, 78 East 56th Street, New York, N.Y. 10022.**

Name________________________________ Amount enclosed $____________
Address ________________________________________________
City________________________ State______________ Zip______________

## PROMqueries

# HAVE A PROBLEM? ASK ROM

### by Eben Ostby

Dear ROM:

A friend of mine, who lives 100 miles away in the next state, has a personal computer, as do I. We would like to interchange programs and data and have the computers interact somehow. Is there an easy way to hook up computers over phone lines?

James McLaughlin
Denver, Colorado

*Dear James,*

*There are dozens of ways to interface computers over phone lines, and some of them are easy and cheap. The easiest, however, are probably not the cheapest. The most flexible method would be to use a* modem *(MOdulator-DEModulator) at each end of the phone line. One end would need the type of modem that answers calls on a time-sharing system; it provides the carrier tone when you connect with it. The other end could use a standard acoustic coupler attached to the RS-232 interface on your computer. This system has the advantage of being standardized and reliable. It is also potentially very expensive. We've seen one modem that would be perfect for this application. The D C Hayes 80-103A Data Communications Adapter, which is ready to plug into your S-100 bus, costs about $280 and is available from D. C. Hayes, Inc., P. O. Box 9884, Atlanta, GA 30319. An old acoustic coupler can probably be picked up for a song or two from your local warehouse sales outlet or computer store.*

*A cheaper though less certain way of communicating by phone involves a modified cassette interface. It may take a lot of experimentation to get it to work (we've never tried), but it should be possible to hook up a Kansas City-type interface to a small amplifier and speaker, and hold the speaker to the phone. At the other end, you'd need a similar interface on your friend's computer. What you'd be doing—if it worked—would be fooling one computer into thinking that it was writing on tape, while it was actually speaking into the telephone. The other end would be fooled into thinking that it was reading a tape. You, the users, would have to be very careful with your protocol. What I mean by this is that you'd have to synchronize your movements carefully, or you might find that your data was not being received. You'll probably have to make a great many trials before you get the bugs off the phone.*

*By the way, if you get it to work, let us know.*

ROM

Dear ROM,

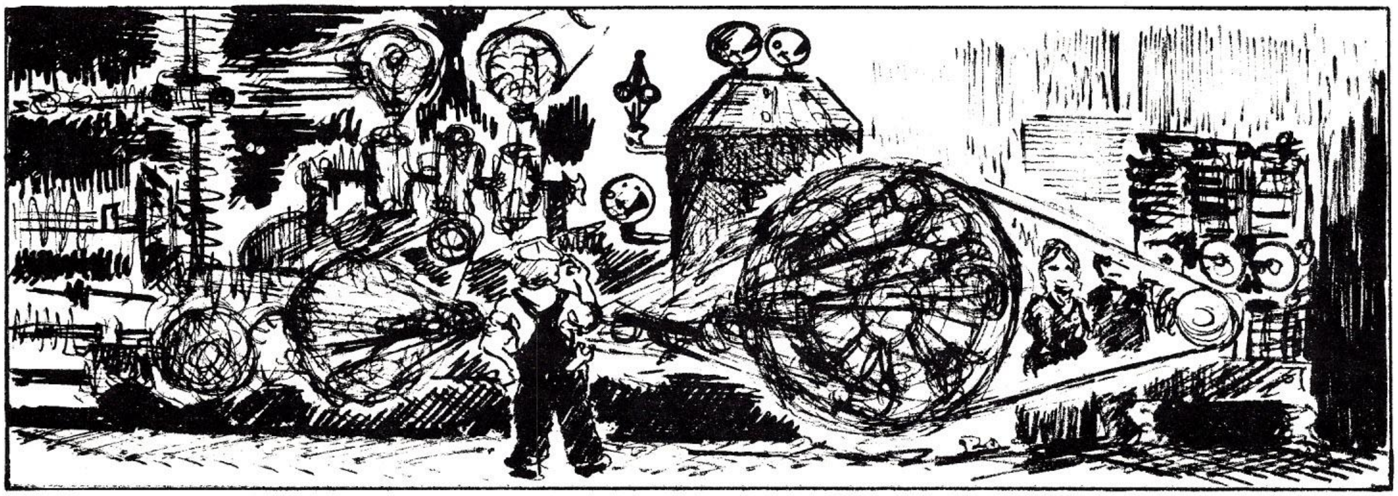
I'm confused about all the different new processing techniques heralded lately in journals such as yours. What are the differences between parallel processing, serial processing, and matrix processing?

Robert Swanson
Minneapolis, Minnesota

*Dear Robert,*

*If you're referring to the way different languages handle data, the differences are really illusory. Even though some languages appear to process arrays in parallel—meaning they appear to operate on all elements of the array simultaneously—most computers actually operate on one element at a time. The high-level software allows you to tell the computer to add the whole array to another one, but the computer ends up doing a separate addition for each element. There is a savings in your own time, though, since you only have to specify one array addition.*

**BABBAGE AND LOVELACE**



*"Watt says it will do 500 rpm."*

Cartoon by Steve Gerling

*Matrix processing is another extension of parallel processing in this sense. Matrix processing is simply the processing of a two-dimensional array in one blow.*

*There are other meanings for the expressions* parallel processing *and* serial processing. *They sometimes refer to the way in which a computer operates on its own data inside the processor (CPU). On a serial machine, additions, comparisons, ANDs, and all other operations take place serially* bit by bit. *Since the hardware designer need only build a one-bit adder instead of an eight-, twelve-, or sixteen-bit adder, the cost of a serial machine is likely to be much less than that of a parallel machine, where all eight bits of a computer word or byte are added simultaneously. A serial machine is also much slower than a parallel machine—imagine eight people each adding one pair of numbers versus one person adding eight pairs of numbers.*

*Parallel processing has reached its apex in the experimental Star computer, which does parallel array operations with hardware instead of software. The Star, in effect, adds many pairs of bytes all at one time.*

ROM

Dear ROM,

So many systems are described as coming with "all the bells and whistles" attached. Where did this expression originate?

Jonathan Strick
Philadelphia, Pennsylvania

*Dear Jonathan,*

*I haven't the vaguest idea.*

ROM

Dear ROM,

I have a research library of approximately 2,000 volumes with roughly 60,000 items I'd like referenced and cross-referenced. Hopefully this would save me more time in the long run than simply entering all the data would. Also I add perhaps 150 journals a year to my collection, each with an average of ten relevant articles.

How big an undertaking would it be, first of all, to develop a program for this? Secondly, from what I've seen, data on floppy discs seems to vanish at the most inopportune times. How do I protect my data once I've got it? And, lastly, is all this even a feasible idea for a microcomputer system?

Ian White
Berkeley, California

*Dear Ian,*

*This sounds like an ideal project for someone with lots of space on a floppy. Actually, two floppies would be more like it. If you figure, say, thirty characters per title, twenty characters per author's name, fifteen characters for a reference number, and another ten for subject category and other flags, each book will require some seventy-five bytes of information—perhaps less if you pack it in tightly. At that rate, you would have 150K bytes of data for your library—much less than a typical full-size floppy holds. But, as you suggest, data stored on a floppy may be less than archival. My suggestion is that you resort to the primitive technique of "backing up" your data, that is, make a copy of your precious files and store it away along with copies of the transactions you entered to modify that copy of the file. This way, if your system bombs and you find out about it (which is no sure thing), you can re-enter the data you entered after you "backed up" the system. In fact, this is a good idea no matter what data you're storing on your disks.*

*I can't, unfortunately, estimate how large a program would be needed to automate your collection. Many factors determine the size of the program for any given task.*

*The number of features your program will encompass makes the biggest difference. You could write a tiny program to read and write records with your book information on them. But you probably want to have sorted lists and searches for books by author, title, subject, and (if you lend books) borrower. As you add this information, both the program and the data space grow proportionately.*

*Facilities for user input tend to be costly in terms of programming complexity. If you restrict the user to typing in a few cryptic symbols for access to data, your input routines, including the ever-present error checking routines, are likely to be fairly small and simple. But if you'd like to be able to ask for a book in simple English, or specify all your options at once, or some such thing, you would have to*

*"But, Babbage dear, what about the glitches?"*

*"Second-generation logic."*

*write a monstrous input routine just to handle possible errors.*

*Packing data often takes a lot of programming effort. "You don't get something for nothing." It is a hard and fast rule that the more compact you want your data to be, the larger and more complex the programs that operate on that data will have to be. All the encoding, packing, decoding, and unpacking routines take up space—and time.*

*Depending, then, on what sort of system you want, and what you have available in the way of memory and peripherals, you should be able to create a very respectable library system. Maybe you could sell it to others like yourself.*

ROM

Dear ROM,

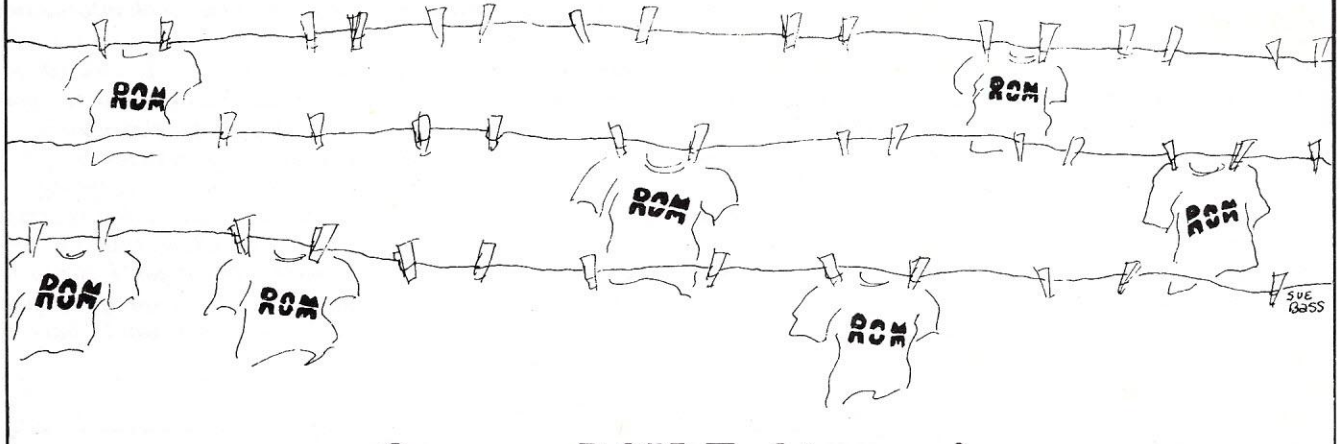
Where I live, we have frequent power failures—not of the New York City variety, but limited, short-term brownouts. Is there a way I can protect my micro in the event of such a failure?

Powerless
Storrs, Connecticut

*Dear Powerless,*

*If you only wish to protect yourself against short-term voltage drops—say down to 90 v—you can buy a voltage regulator, a box which plugs in between your computer and the wall. It won't help if the power goes off for a second, though—then you'll still lose your data. If this is likely in your area, you might try running the computer on batteries. If that isn't feasible, look for a computer attachment that will cause an interrupt when power levels drop. This attachment monitors the line voltage, and when it drops below a certain level, the device issues an I/O interrupt to the processor. Such devices were common on minis some years back although lately better back-up power systems have made them less necessary. If you were to design a low-power interrupt system for, say, an 8080-based machine, the power monitor would trigger the interrupt in the computer when the power dropped below the level necessary to maintain proper system performance. The monitor would also issue a single computer instruction to the processor, which would execute that instruction immediately. Usually, a subroutine jump instruction (RTS on the 8080) transfers program execution to a routine that quickly puts the system into a state where only a minimum amount of data can be lost. If time permits—you would probably have a few milliseconds before the power was too low for operation—your routine can try to dump important parts of memory to a floppy. The whole idea is not to avoid the unavoidable but to make restoring the system as easy as possible when the lights come back on.*

ROM



**Get your ROM T-shirt now!**

**Order from:**
**ROM Publications Corp.**
**Route 97**
**Hampton, CT 06247**

**Please ship to:**
**Name** ____________________
**Address** ____________________
**City** __________ **State** __________ **Zip** __________

**☐S Qty.** ______
**☐M Qty.** ______
**☐L Qty.** ______ **☐Child S Qty.** ______
**☐XL Qty.** ______ **☐Child L Qty.** ______
**☐Check or money order enclosed.**
**☐Master Charge ☐BankAmericard**
**Exp. date Card#** ____________________

**$5 ppd.**
**2 for $9 ppd.**

**Please allow 4 to 6 weeks for delivery.**

## Legal ROMifications

# GETTING AROUND THE POST OFFICE

### by Peter Feilbogen, attorney at law

New technology brings with it a host of peripheral businesses and at times eliminates or expands others. One business which has expanded is messenger service. But in servicing the data-processing industry, messenger services have run afoul of the law.

The United States Postal Service has a monopoly on the delivery of mail. However, the United States Postal Service does not offer same-day delivery nor does it guarantee next-day delivery of parcels. Thus, a regulation was adopted specifically for data-processing materials; it may be found in Title 39 of the Code of Federal Regulations, Part 320, entitled "Suspension of the Private Express Statutes."

The benefit that this exemption bestows is that the sender is not obligated to pay postage to the Postal Service over and above the delivery fee charged. There are, of course, rules which must be followed:

1. Only data-processing materials sent between the data-processing center and the office originating the data-processing materials are covered.

2. The delivery must be completed within twelve hours or by noon of the addressee's next business day. Data-processing work must be commenced on any such material sent to a data-processing center within thirty-six hours of receipt. Shipments between a domestic point and a foreign point shall be deemed to begin at the time materials of foreign origin are received at the international gateway city or end at the time materials of domestic origin leave the international gateway city.

3. "Addressee's next business day" means the first calendar day, stated in his local time, on which he conducts business, following the calendar day of dispatch.

4. "Data processing" in the context of the Regulation means electromechanical or electronic processing.

5. Data-processing "materials" for this part include materials of all types that are ready for immediate data processing or for automatic conversion into a form ready for immediate data processing, as well as the direct outputs of data processing, but only if they are produced on a regular periodic basis.

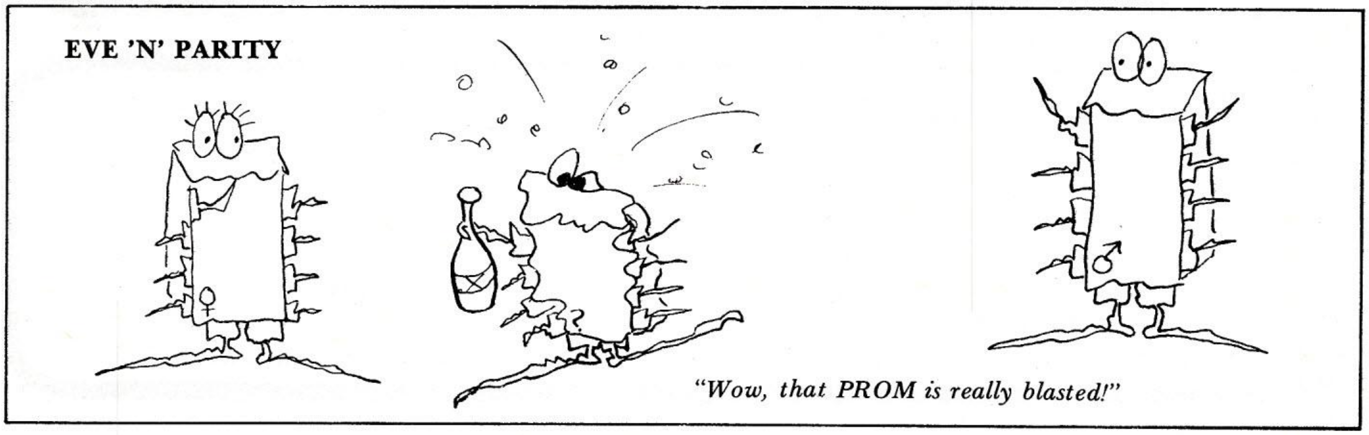
6. Those carriers involved in such operations must keep detailed records of operations, both in dollars and in volume. These records provide documentation in two areas. First, the Postal Service is entitled to inspect such records to determine that the carrier operations are within the parameter of the suspension. In the second place, the Postal Service reserves the right to revoke this suspension. However, the carrier may maintain operations at a level existing at the time of the revocation in the particular market served prior to revocation. As a condition to continuing operations, the carrier must be able to provide accurate data supporting the claimed volume.

7. To establish operations under this suspension, notification must be submitted to:

   PRIVATE EXPRESS LIAISON OFFICER
   CUSTOMER SERVICES DEPARTMENT
   UNITED STATES POSTAL SERVICE
   WASHINGTON, D. C. 20260

   on a form available from the Private Express Liaison Office.

It is the responsibility of the carrier to see that the material it transports conforms with the statutes, that the container and covers are properly marked, and that actual times of pick-up and delivery are maintained. ▼

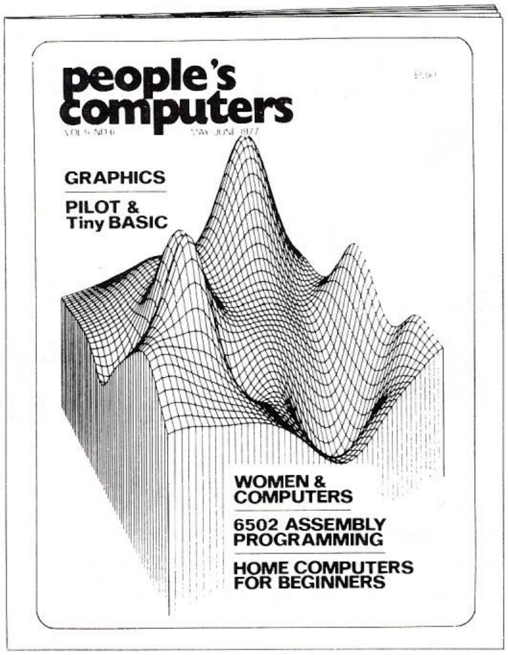


**EVE 'N' PARITY**

*"Wow, that PROM is really blasted!"*

Cartoon by Korkie

# WE'RE DIFFERENT!


people's computers
VOL 5 NO 6
MAY-JUNE 1977
GRAPHICS
PILOT & Tiny BASIC
WOMEN & COMPUTERS
6502 ASSEMBLY PROGRAMMING
HOME COMPUTERS FOR BEGINNERS

**No Ads**

**No Frills**

**All Content**

*People's Computers* is for anybody and everybody who wants to learn more about computers, how they work and how to use them. We aren't padded with color advertisements, we're cover to cover with articles, listings, reviews, games, letters and interviews.

Read the whys and wherefors of different computer languages. Discover what a 'chip' is, how computer networks operate and what new products are available. Learn to program from our series on easy-to-use computer languages. Find out how computers are used in education, in robotics, and for communication by the handicapped. Each issue also contains programs with extensive comments, that you can use on your computer at home, at school or at work.

# people's computers

Please start my one year subscription (6 issues) to *People's Computers* and bill me for $8.

NAME ______________________________

ADDRESS ______________________________

CITY ______________ STATE __________ ZIP __________

Mail to: *People's Computers*, Dept 52, 1263 El Camino Real, Menlo Park, California 94025

## AIQuotient

# PERSPECTIVE ON PERCEPTION

### by Stuart Dambrot

Have you ever seen a "baby" Volkswagen? How about one that is "elderly"?

Robert Shaw's subjects did.

Robert Shaw, a psychological researcher at the University of Connecticut, has derived a mathematical function—the aging transformation—and developed computer software surrounding it. The program first generates a profile of a photographed object (e.g., a human skull); the transformation is then applied to the profile, and can be run either forward or in reverse.

In an experiment, the responses of subjects viewing a forward transformation indicated that they were perceiving incremental aging of the profile. When the programs were reversed, the subjects saw the profile "getting younger." This held true for profiles of both human beings and Volkswagens.

The purpose of Shaw's experiment was to provide evidence for the existence of "affordances," a concept generated by J. J. Gibson of Cornell University. In a preliminary version of *An Ecological Approach to Visual Perception,* Gibson defines an affordance as "a combination of physical properties of the environment that is uniquely suited to a given animal." For an adult human being of average height and weight, a horizontal surface with a minimal area of about one square foot, approximately two feet above the ground, and of reasonable substantiality specifies the *meaningful* affordance "sittable-uponness"; the surface says "sit on me."

This affordance is perceived only when the person intends to sit; it exists, however, *whether or not* it is being discerned. The person and surface are related by the *activity* of sitting. Three entities—meaningful affordance, the organism's intention, and the activity that relates them—form an ecological *event.*

Consider a man who is salt-water fishing in a small boat. (The luck he's having is irrelevant.) After a while, he becomes hungry. Glancing about, he espies an apple next to his tackle box; assuming he doesn't despise apples, what happens next is obvious.

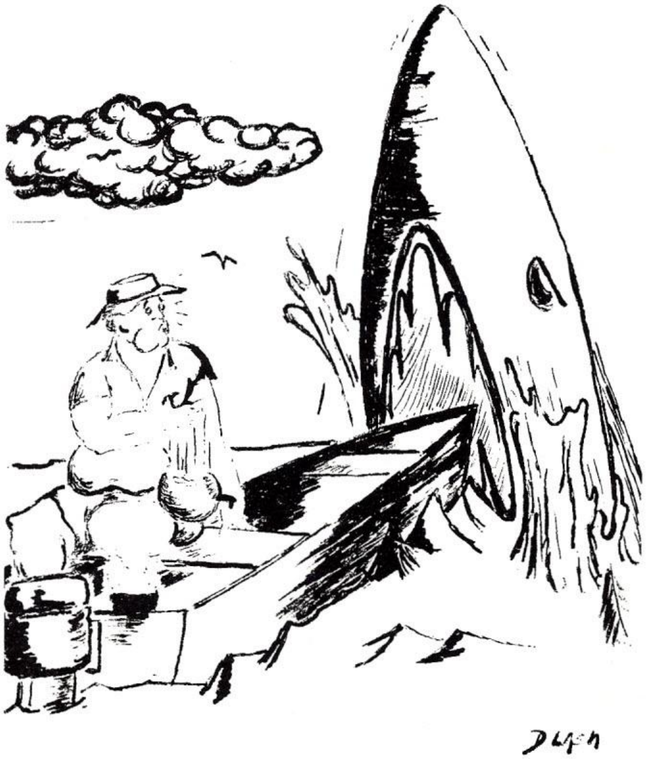
The fellow's *intent* is to eat; the apple's *meaning* is its edibility for human beings; eating, of course, is the relating *activity.* But what if our hypothetical fisherman were unfortunate enough to notice a large, rapidly approaching shark? The event would still be one of a consummatory nature, but the affordance perceived by our poor friend would certainly be quite different.

The theory of affordance is a radical change in the assumptions of psychology—maybe the only such change since the times of Plato and Aristotle. If this seems a bit extreme, reflect for a moment on the "elderly Volkswagen" phenomenon. What Shaw's subjects saw when the aging transformation was applied to Volkswagen profiles strongly suggests that affordances actually exist. Asked to explain this, most of us would say that we have an idea about what old things tend to look like; the same holds true for very young things.

Let's go a little further. Where do these ideas come from? Usually, only two possible sources come to mind: either we are born with such ideas, or we derive them from experience. The first explanation is a *nativistic* one; the second belongs to a point of view known as *empiricism.*

Nativism is the doctrine of innate, or inborn, ideas. Basically, it states that there are inherited items in human knowledge which are in every individual, regardless of experience. Empiricism, on the other hand, is characterized by the notion of the *tabula rasa,* or *blank slate;* the *sole* source of knowledge is what experience etches on the slate, which is completely blank at birth. Further, the empiricist defines *mind* as the set of all associations made between pieces of sense data. Both share the following assumptions:

- Information about objects in the environment (light reflected off a surface, a particular odor, a certain sound, etc.) is meaningless in and of itself.
- Animals (including man) seem to have suddenly appeared in a world in which they are able to survive; the interactive nature of evolution is grossly underestimated.



Illustrated by David Lash

Pondering these definitions and premises, one finds some unsettling inconsistencies. How can a nativist account for the connection that an individual has to make between meaningless sense data and the innate idea that allows recognition to occur? Can an empiricist reasonably claim that meaningful associations can be made from meaningless sense data?

Maybe the resolution lies in environmental information which is rich in meaning. Gibsonian psychology is the only theory of perception and knowledge with this assumption.

If Gibson is right, his affordance theory clearly points to modification in the premises of many fields, notably philosophy, psychology, computer science, and artificial intelligence.

Artificial intelligence can be defined as the design or capability of a machine that can perform functions normally associated with human intelligence: learning, reasoning, automatic error connection, gathering meaning from context, and so on. Artificial intelligence, then, is obviously subject to the influence of affordance theory.

The main thrust of artificial intelligence research has been confined to the development of extremely sophisticated software, with many notable results. Affordance theory strongly suggests that an evolutionary approach would prove fruitful, and there have been endeavors in this vein; the process is termed *simulated* evolution, because the "organisms" involved are software systems.

Several devices based on artificial intelligence principles are available to the computer hobbyist. Most of these are concerned with speech recognition and production. An example of the former is SpeechLab by Heuristics, Inc. SpeechLab has a potential sixty-four-word vocabulary which it "learns" using a statistical algorithm. This algorithm is a relatively simple one, but it demonstrates several fundamental concepts of speech recognition. During "training," a word is analyzed and translated into digital code. A *template,* or model, of the word is then generated from this code and stored in memory. Several samples are taken of each word so that the natural variations occuring in spoken language are incorporated into the template. (Just try repeating a two- or three-syllable word ten times with no variation. It's pretty damn hard—especially if the intervals between repetitions are relatively long.)

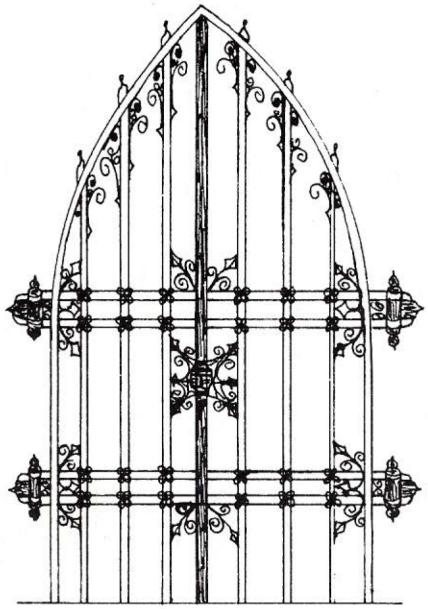
When SpeechLab is used as an input device, a spoken word is digitized and compared to all the templates; recognition is defined as the closest match found during these comparisons. Once this happens, the routine specified by that template is begun—e.g., PRINT, STOP, DIAL (yes, for a telephone), and various numbers.

VOTRAX, COMPUTALKER, and AI Cybernetic Systems Model-1000 Speech Synthesizer are small-system oriented machines that *generate* human-like speech. Digital versions of phonemes (the basic sound units of human language) are permanently stored in memory; these are electronically synthesized under software control to produce understandable words and phrases.

"UH, STUART?"

"Yes?"

**OPEN A GATE FOR SOME FRIENDS**



**Send a gift subscription to ROM today**

ROM
COMPUTER APPLICATIONS FOR LIVING

**Please send a gift subscription to:**
**Name** ______
**Address** ______
**City** **State** **Zip** ______

**United States: ☐One year $15 ☐Two years $28 ☐Three years $39**
**Canada and Mexico: Please add $2 per year additional postage.**
**Europe and South America: Please add $12 per year additional postage.**
**All other continents: Please add $24 per year additional postage.**
**☐Check or money order enclosed ☐Master Charge ☐BankAmericard**
**Exp. date Card#** ______

**Name of sender** ______
**Address** ______
**City** **State** **Zip** ______

**☐I would like a gift card enclosed with the following message:** ______

**ROM Publications Corporation, Route 97, Hampton, CT 06247**

"YOUR WATER HAS BEEN BOILING FOR ONE HOUR AND SEVENTEEN MINUTES. I ESTIMATE FOUR MINUTES AND THREE SECONDS TO IRREVERSIBLE UTENSIL DAMAGE."

"Oh, sure. Thank you."

"YOUR LAST STATEMENT IS AMBIGUOUS."

"What? Oh, yeah. Stove burner off."

"ACKNOWLEDGED."

While such devices are fascinatingly useful, they are obviously quite limited. Even the most advanced machinery available cannot do that much better. Why is this? Perhaps it's because their design is derived from research that is based on those nativistic or empirical assumptions we mentioned earlier. Note how similar SpeechLab's template algorithm is to the process of matching supposedly ambiguous sense data with a "correct" innate idea. Both assume that world information is meaningless, and therefore can be understood only by virtue of complex intermediary procedures. Is it reasonable to say that the same organism that invented statistics actually goes through such computations while interacting with the world?

Not on your life.

The development of intelligent machines has been hampered by the belief that the environment does not contain well-specified and meaning-rich information for the organisms it supports. We assume that meaning must be *inferred* from the world through a careful analysis of physical qualities—mass, color, luminosity, and so on. Hence the emphasis on the software simulation of intelligence. But what about hardware? Can materials be constructed that could respond *directly* to affordances?

At a New Jersey Bell Telephone Laboratory there have been some rather encouraging findings in this direction: scientists have created a hitherto unknown crystalline substance by successively depositing atomic layers of specific chemical elements. The new semiconductor has unexpected properties within which context hardware evolution becomes a more viable concept.

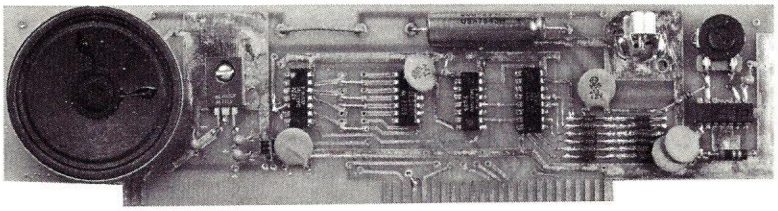
A technique of this type could have fantastic implications. One can envision a collection of "electronic organisms" with inherent functional capabilities. One "species" might evolve to the point of having some type of "intelligence"; in that case, the term *artificial intelligence* would no longer be appropriate. Fields such as *electronic biology* and *analogous intelligence* might appear.

On a less exotic level, it is tantalizing to imagine the possibilities in the area of personal computing: natural language understanding, processing, and production; a new generation of truly interactive games; sophisticated security systems; even astoundingly versatile monitor and control applications.

At the most basic yet profound level, affordance theory asks us to re-examine the way in which we think about human thought. Our tools are extensions of ourselves; computing devices are cognitive tools, and what they are like is a function of how we view human perception and knowledge. ▼

NEWTECH
Model 6

**MUSIC BOARD**



**PRODUCES MELODIES, RHYTHMS,
SOUND EFFECTS, MORSE CODE,
TOUCH-TONE SYNTHESIS, AND MORE!**

- S-100 bus compatible
- Jumper selectable address decoding
- 6-bit latching digital-to-analog converter
- Glass epoxy printed circuit board with plated-through holes and gold-plated fingers
- Audio amplifier
- Speaker

**$59.95 ASSEMBLED AND TESTED
AVAILABLE THROUGH YOUR
LOCAL COMPUTER STORE**

- Volume control
- RCA phono jack for connection to external audio system
- Complete users manual with BASIC program for writing musical scores and 8080 Assembly Language routine to play them
- 60-day parts and labor warranty

NEWTECH COMPUTER SYSTEMS, INC.

131 JORALEMON STREET * BROOKLYN, NEW YORK 11201 * (212) 625-6220

# PROJECT PROMETHEUS

*Solar energy is the "in" topic of the fuel-short seventies. One way or the other, it's in the news almost weekly, either because it can't be done or because someone has just invented the closest thing to a perpetual motion machine since molecular activity first began—in his basement, of course. Even the government is getting into the act, tossing a few million here and there for feasibility studies or development programs to show that it really is thinking past the next election, or at least up to it.*

*Once blinder-ridden bureaucrats and industrialists have made and lost millions pushing nuclear-based energy beyond all reason and safety, solar energy will come into its own. Of that there seems to be little doubt—at least if civilization survives its current self-destruct mode. What form the energetics of Father Sol will take, however, is open to a great deal of speculation.*

*In my own opinion, which is not only limited to being an opinion but is also a wish whose realization lies several decades away, the most efficient solar energy conversion systems to be up and running will be photosynthesis-based emulations of plant life.*

*Nature, after all, has a lot more on-hand development experience along the lines of energy production than mankind does. Given our monumental egos, which in themselves make many impossible things possible, it is still unlikely that we will nullify Zeno's paradox. But if it is improbable that our fragmented technology will catch up with the efficiency of the fully integrated energy clocks of nature, it is true that we will probably make our greatest strides by paralleling our research with that already finely tuned by evolution. Nevertheless, biobased energy conversion is really something for the next century.*

*A little closer to our time frame are huge solar collectors in space, beaming their energy down to earth. They have several advantages; for example, focusing these energy beams on specific targets makes possible at last that pet military toy, the death ray. It would be difficult to foresee any government passing up the chance to underwrite such a Damoclean sword—unlimited cheap energy and unlimited cheap destruction with no loss of life to the sponsoring side would most certainly be one of the twentieth century's great vote getters.*

*A second appeal of space-based solar energy collectors is that they get us into a rather hot kettle of fish, considering the third law of thermodynamics. By collecting energy that would normally bypass the earth, and beaming it down to what is essentially a closed system, we're doing the same thing as turning the burner on under the coffee pot. Things are going to get hotter. All of which will make a great disaster movie, with the polar ice caps melting, New York City at last getting its streets washed clean.... What it probably won't do is affect our generation much, being in a time frame that excludes us.*

*As for now, solar energy offers the opportunity to start bringing the fireplace back home—discrete energy production at the family level, efficient energy production at the user's level. Project Prometheus is hopefully a small step in this direction.*

*Energy production necessitates, as a first step, energy measurement. The question of how much energy you have available must perforce be answered before you can determine the needed method and conversion to produce the required final output. To that shining end we present Project Prometheus to help you and your micro plan for a solar future.*

*On a grander scale, we hope in a year or two to have enough participants (computer clubs out there, are you listening?—here's a practical far-ranging project for your micros) to be able to develop the first rough empirical base line of solar radiation across the United States. This would be by necessity a volunteer effort and we'd appreciate as much feedback and as many suggestions as you can offer.*

*—The Editor*

## Going Solar with Your

No one can lock up the sun and put a meter on it—it's free for the taking. But the very diffuseness of solar energy, its great advantage, is also its disadvantage. For in order to make efficient use of solar energy, we can't apply the same thinking habits that we have learned from using fossil energy sources. We can't rush down to the hardware store and buy a gizmo which will supply us with unlimited power from an occasional exposure to sunlight. The concept of unlimited expansion of energy consumption does not apply to solar energy. If we want the benefits of solar energy, we will have to measure its availability accurately, calculate our intended usage realistically, and control our collection, storage, and usage systems carefully.

A typical pronouncement on the feasibility of solar energy usage today

Photograph and all diagrams by the author
Pencil drawing by Rex Ruden

# Micro

by
Lee
Felsenstein

seems to be that in twenty or thirty years the technology will be sufficiently developed so that the sun can take its place as a "significant" source of energy. Upon examination, "significant" appears to mean "available through high-technology systems which allow and require its concentration and sale for use with no change in energy consumption patterns." The same people who make this kind of statement seem also to believe that computers are million-dollar room-sized accounting machines which will only become bigger and more expensive.

Examination of the assumptions on which such pronouncements are made would seem to indicate that small-scale energy control and measurement technology has not progressed beyond the introduction of the mechanical thermostat in 1930. At the same time, examination of the facts leads to the conclusion that the only area in the field of solar energy collection in which sophistication is even possible is in the field of measurement and control. The basic principles of solar energy are, after all, almost trivial:

1. If you put something in direct sunlight, it will tend to get warm.
2. A large mass, when warmed up, will take a long time to cool down.
3. Most fluids get lighter when warmed and will rise.

We may not know the reasons why these things happen, but the mathematical equations which describe this behavior are fairly simple. No one expects any technical breakthroughs which will fundamentally alter these principles or which will discover new ones.

In the realm of measurement and control, however, it may be that the requisite twenty or thirty years of technology development predicted by the experts has already occurred. A microcomputer can do the drudge work of taking solar measurements for days, weeks, and months at a stretch, so that a prospective user can answer the question of how much energy is available at a given location. The computer can measure several variables and can even perform checks of its own behavior, so that sudden shifts or wide swings which might indicate malfunction can be spotted and flagged by the equipment itself. The ability to reduce data immediately, "in real time," means that the measurement system can be incorporated into the control system by means of slightly more complex programming and the connection of relays and valves as output devices.

We don't have to wait. We can start now using our microcomputers for our own individual and collective benefit in completing the development of solar energy.

*A microcomputer makes an excellent drudge, patiently taking the measurements you want for days, weeks, months at a stretch.*

In the design of a solar heating system the primary unknown is usually the amount and time distribution of sunlight available at the proposed site. Measurements of solar radiation are being made at a number of locations around the country by the National Weather Service and by several universities, but these measurement systems are carefully set up to exclude effects such as reflections or shadows from buildings. They provide a good basis for calibration of other measurements, but they don't tell you how much sun comes in to your specific location.

These measurements must be corrected not only for the less-than-perfect horizons which exist at a given installation site, but also for its microclimate: the localized variations in sky conditions such as fog, clouds building up over mountain ranges, industrial smoke plumes, etc. There are two ways of approaching such a problem. One, the analytical approach, starts from data taken under idealized conditions and applies correction factors to simulate the real conditions. The other, the empirical approach, starts by measuring data under the real conditions. It is important to note that if the analytical approach is used, the correction factors must still come from empirical measurements.

There is a great shortage of empirical information in solar data. Even the most official measurements of insolation, as incoming solar radiation is called, have suffered from deterioration of the measuring apparatus, and much of the currently collected data is being "rehabilitated" by the National Oceanic and Atmospheric Administration (NOAA), which operates the National Weather Service. Instruments now in use are capable of accuracy to about five percent, and many armchair prospective data users would like to see two percent or one percent.

When it comes to measurements of practical value to a user, however, a lot of the hair-splitting stops. Everyone pretty much agrees that information taken at your present or future site with a pickup oriented as the collector will be tilted and accurate to about ten percent is quite sufficient for the design of a solar energy system. It's only when you decide to put your data into a handbook for general use that problems start. Then the pickup must be accurately levelled, mounted with a clear horizon, free from vibration, and on and on.

Project Prometheus is aimed towards participants wishing to set up measuring stations for empirical data of local value only. As skill and equipment improves, those who wish may proceed to become involved in handbook-quality data collection for wider use.

The measurement of insolation for heating purposes is simple in principle: put something painted flat black out in the sun and see how it heats up. In practice, it is not quite so simple. A device for such measurement is called a pyranometer, and precision pyranometers for scientific-grade data cost about six

## A POCKETFUL OF PICKUPS

Some pyranometer-type pickups are kit, some are homebuilt, others are fully built and calibrated. The sampling given here is for your information only and does not constitute an endorsement of one pickup over another.

An article entitled "Measure The Sun's Energy with A Solar Radiometer" by Warren Jochem in the December 1976 issue of *Popular Electronics* magazine describes a solar cell with a shunt load and a meter which is easily convertible to readout by an A-to-D converter and is calibrated.

Edmund Scientific Co., Barrington, NJ 08007, has an electronic pyranometer (called a Sun Meter) which is calibrated and similar to the *Popular Electronics* unit. Edmund also has weather instrumentation such as anemometers and rain gauges with digital and analog electronic outputs. Their latest catalogue contains a fund of details and much helpful information.

Dodge Products, P.O. Box 19781, Houston, TX 77024, sells an assembled and calibrated silicon pyranometer for about fifty dollars.

Rho Sigma, 11922 Valerio Street, North Hollywood, CA 91605, sells pyranometers as well as other environmental measurement instruments.

### The One We Picked

Following the design in the *Popular Electronics* article, we bought a Calectro J4-800 solar cell and the recommended meter. With an 0.5-ohm resistor in parallel with the meter (which had a resistance of 1.7 ohms) we observed about 50 millivolts per Langley per minute (one Langley is one calorie per square centimeter, equal to 3.69 BTU per square foot). This was not calibrated but was taken from the calibration suggested in the construction article.

The resistance load presented to the solar cell in the article was just under 0.4 ohms; with an 0.36-ohm resistor connected across the solar cell (which was mounted within its clear plastic shipping box), the voltage produced at about 2:00 P.M. on an August day was about 35 millivolts.

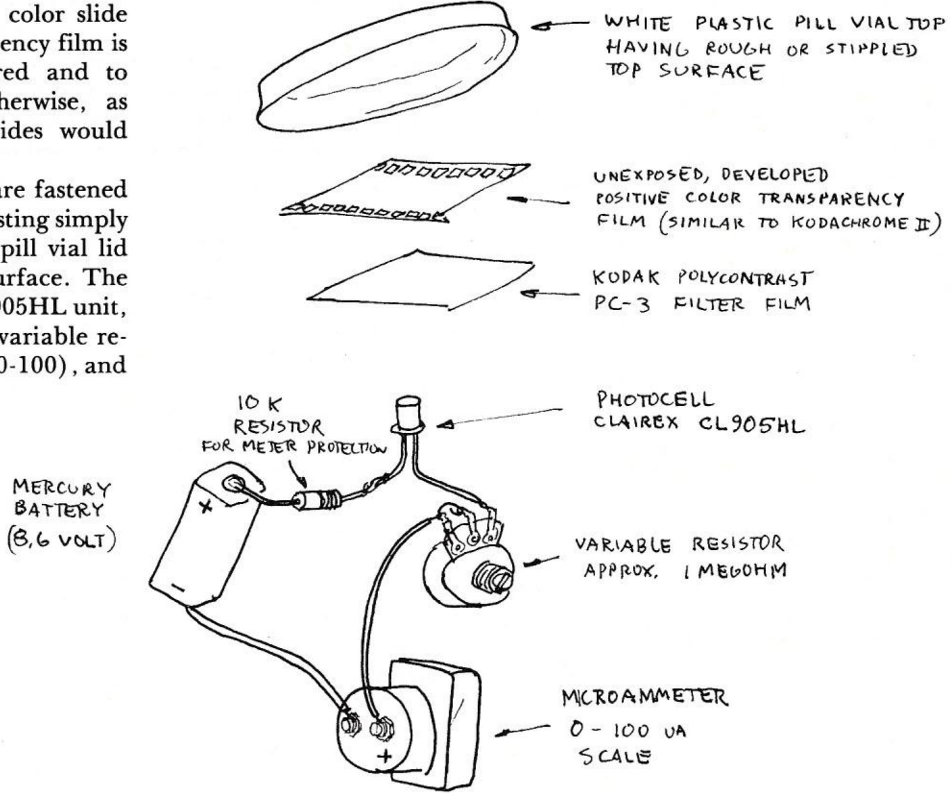
Other solar cells (those of other manufacturers and of different sizes) will result in other voltages. In all cases the load resistance should be kept to less than 0.5 ohms, since we are attempting to measure short-circuit current out of the cell. Current produced under these conditions is quite linear with respect to the solar radiation received.

# PYRANOMETER PICKUPS—HOMEMADE STYLE

Raymond Bahm, Director of the New Mexico Solar Energy Resource Assessment Project at the University of New Mexico, recently displayed a small battery-operated pyranometer pickup using a cadmium sulfide photocell. Since the response of such a photocell is generally limited to a band of visible light, Ray compensated for its low infrared response with a simple filter arrangement: a single sheet of Kodak Polycontrast PC-3 filter material together with a sheet of unexposed but developed Kodachrome II positive color slide film. The color transparency film is designed to pass infrared and to block visible light; otherwise, as Ray points out, the slides would burn up in a projector.

The two filter sheets are fastened to a diffusing cover consisting simply of a white polyethylene pill vial lid with a roughened top surface. The photocell, a Clairex CL905HL unit, is wired in series with a variable resistor, a microammeter (0-100), and a mercury battery. The calibration is relative, set by the variable resistor, which should be in the megohm region.

A circuit such as this can be built easily, and it can be "cut and tried" as far as battery voltages and resistor values are concerned. The homely little pyranometer can serve quite well as a secondary pickup working along with a silicon pyranometer.


WHITE PLASTIC PILL VIAL TOP HAVING ROUGH OR STIPPLED TOP SURFACE
UNEXPOSED, DEVELOPED POSITIVE COLOR TRANSPARENCY FILM (SIMILAR TO KODACHROME II)
KODAK POLYCONTRAST PC-3 FILTER FILM
10 K RESISTOR FOR METER PROTECTION
PHOTOCELL CLAIREX CL905HL
MERCURY BATTERY (8,6 VOLT)
+
VARIABLE RESISTOR APPROX. 1 MEGOHM
MICROAMMETER 0 - 100 UA SCALE
+

CADMIUM SULPHIDE RELATIVE PYRANOMETER. THE METER MAY BE OMITTED IF THE OUTPUT IS TO BE ELECTRICAL, A-TO-D CONVERTER MAY BE CONNECTED TO MEASURE THE VOLTAGE ACROSS THE PHOTOCELL.

hundred dollars at present. They have double Schott-glass domes for control of heat loss, and in spite of that they still are prone to errors from internal reflections or misorientation.

A silicon photocell, on the other hand, can serve as a linear pyranometer pickup when heavily loaded (one-ohm load). It misses some of the infrared spectrum, but this is not fatal, since the error it introduces is predictable and will tend toward conservative design in collectors. Calibrated silicon pyranometers, currently used in agriculture to help estimate irrigation requirements, are available for between fifty and a hundred and fifty dollars.

Cadmium sulfide (CdS) photocells can also be used for insolation measurement if compensated with filters. And a homemade pyranometer using a CdS cell is not difficult to construct.

## ROMtutorial ROMtutorial

*Microcomputer:* A small computer using a microprocessor for its central processing unit. A microprocessor is a CPU, or central processing unit, built using large-scale integration.

*Real time:* Descriptive of on-line computer processing systems which receive and process data quickly enough to produce output to control, direct, or affect the outcome of an ongoing activity or process.

*CPU:* Central processing unit. The "thinking" portion of a computer. It decides where to move information, what to do when it's there, and where it should look for its next instructions.

*I/O:* Input/output. The electrical channels through which the computer moves information to and from the outside world.

*Analog-to-digital converter:* An electronic device that looks at a voltage which can have a range of different values and converts it to a code consisting of digital on-or-off signals.

*Bit:* The simplest unit of information in computing—the condition of being either on or off. Electrical circuits are well-suited to dealing in bits, since it's no problem (usually) to tell the difference between a high and a low voltage.

*Software:* The programs that are run on a computer. A computer system consists of hardware—the computer and its accessories—and software—the programs which make the hardware work the way you want it to.

*Volatile memory:* The internal storage memory of the computer itself, as opposed to tape or disk storage media, which are nonvolatile. Most home computers have volatile memory.

*RAM:* Random access memory. Memory like a set of pigeonholes, into any of which the computer can put new information or from any of which it can read old information. The computer can choose any pigeonhole (or address) at any time. RAM can store and recall information at high speeds and permits information stored in it to be changed at any time.

*Serial interface:* An interface is the dividing line between two electronic devices. A wire or cable usually goes across an interface carrying electrically coded information. If the information moves in a sequence through a single wire, the interface is serial.

Besides the pyranometer pickup or transducer—and apart from the CPU, memory, and I/O that any computer requires—the minimum equipment necessary for insolation measurement includes an analog-to-digital converter of sufficient precision and an electronic clock module. "Sufficient precision" means having enough bits of resolution to allow accurate measurement within the limits which you find acceptable. One percent error means seven bits (1 part in 128) resolution. Each bit less doubles the percentage of error. An error percentage of 6.25 would be one part in sixteen, a degree of precision available from a four-bit DAC (digital-to-analog converter).

Alert or experienced readers will have noticed that we have just mentioned a digital-to-analog converter when talking about conversion in the other direction, from analog to digital. We have done so because both converters use a D-to-A element. In the case of the A-to-D conversion, we include a comparator which looks at the analog signal from the D-to-A converter and compares it with the input voltage. The comparator returns to the circuitry a digital signal which indicates "greater" or "lesser." Using this response, the hardware or software can then narrow in on the final value.

A clock is a necessity in the system if one is to take practical advantage of the computer's capabilities. Those who have spooled through yards of chart recordings trying to "reduce the data" to a manageable and meaningful table know what a blessing it would be if the machine dispensed with the intermediate data and told them only the data they wanted to know. This might be a list of times and temperatures, marking the points at which the temperature had changed by more than two degrees from the last reading. Or it might be a table of temperature readings at regular time intervals. Or it might be a combination of the two so that sudden wide temperature swings could be captured along with the small, slow changes. Ask someone who has had to match up two chart recordings from machines whose paper drive motors have been running at different speeds whether this kind of data collection is worthwhile. Be prepared for an emotional reaction.

*There is still a great shortage of empirical information in solar data.*

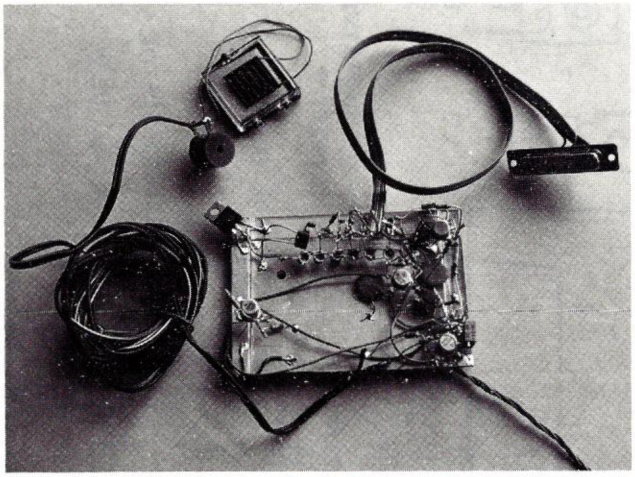
Another advantage of involving a computer in collecting and sorting insolation data is that the data can be stored in a small amount of memory, since it is reduced before it is stored. Even so, the data should not be entrusted to volatile memory (like semiconductor RAM) for any appreciable length of time when there is a chance that a power failure will erase it or that the program will over-write it. The data must regularly be moved off to a nonvolatile storage medium.

The simplest such medium is writing on paper, transcribed by the operator reading memory locations through the front panel switches and lights. Or the data could be printed out on a Teletype or similar printing terminal through a serial interface. If an ASR machine is available, the data can be punched onto paper tape as it is listed out. This makes it machine readable in case it is to be processed further by the computer or transmitted to another computer.

If your computer has a cassette tape interface, that should be used to store the data at regular intervals. Either the intervals can be automatically timed out by the clock reading or they can be at your convenience when you don't want to commit a tape recorder to the system continuously.

In setting up your pyranometer for insolation measurement, it's a good idea to put a little time and effort into mounting the pickup so that it will be sturdily braced and well protected from the elements. You want it to yield reproducible results. To facilitate accurate positioning, the pickup should be mounted on the front surface of a piece of plywood or particle board and shimmed or otherwise adjusted so that the surface of the pickup is parallel to the surface of the board. Once these critical adjustments have been made, it becomes a much simpler task to mount the board parallel to the intended plane of the collector with accuracy and repeatability.

Your pyranometer pickup may be kept in calibration by periodical checks, and a preliminary calibration



# A HOMEBREW ANALOG-TO-DIGITAL CONVERTER

Shown on the following page is the schematic of a successive-approximation analog-to-digital converter designed and built by the author. It is built with standard non-precision components and is capable of five bits of resolution, or one part in thirty-two. The input amplifier is scaled for voltages in the region of 100 millivolts to 1 volt, but a change in the feedback resistor allows other scalings.

The 710 comparator looks at the amplified input voltage (which has been inverted so that a positive input voltage yields a negative amplified voltage) and the voltage generated by the digital-to-analog converter. The characteristics of the 710 dictate that these voltages not differ by more than 5 volts (other comparators are not necessarily subject to the same restriction), and this sets the maximum amplification of the input amplifier and the scale factor of the D-to-A converter. Both factors are set by the feedback resistances in the operational amplifier stages.

A range of 0 to −3.10 volts was chosen to allow easy calculation and to stay within the 5-volt limit. The comparator returns a "high" logic level when the input voltage is below the trial voltage. A small amount (4 millivolts) of hysteresis has been provided at the comparator inputs to prevent oscillation and response to noise. The data inputs are "high active," that is, a high logic voltage level represents a "1."

The circuit works from unregulated plus and minus 17-volt supplies. These voltages are fed directly to the operational amplifiers. The 710 requires +12 volts regulated and −6 volts regulated. These are supplied by a 7805 integrated regulator and a 5.1-volt zener diode respectively. The negative terminal of the 7805 is connected to the output of another 7805, which in turn supplies 7 volts to the current source transistors. The 7805 provides 5 volts output referenced to its negative terminal, and the negative terminal of the 7-volt regulator goes to a chain of three diodes, one of which is created by the base-emitter junction of a transistor. The purpose of this transistor is to imitate the temperature-dependent variation of the current source transistors. Physically it should be in close proximity to the other transistors.

The 2.1-volt drop of the diodes boosts the output voltage of the 7805 regulator to 7 volts. This voltage in turn boosts the output of the second 7805 to 12 volts to feed the 710. A 723-type voltage regulator may also be used to supply the 12 volts. One happened to be handy when this circuit was built.

The circuit was constructed on a piece of single-sided laminate (fiber glass with a copper coating). Areas of the copper were separated by shallow saw kerfs. It is always a good idea in A-to-D converters to allow large ground planes to reduce "crosstalk" noise between the analog and digital sections.

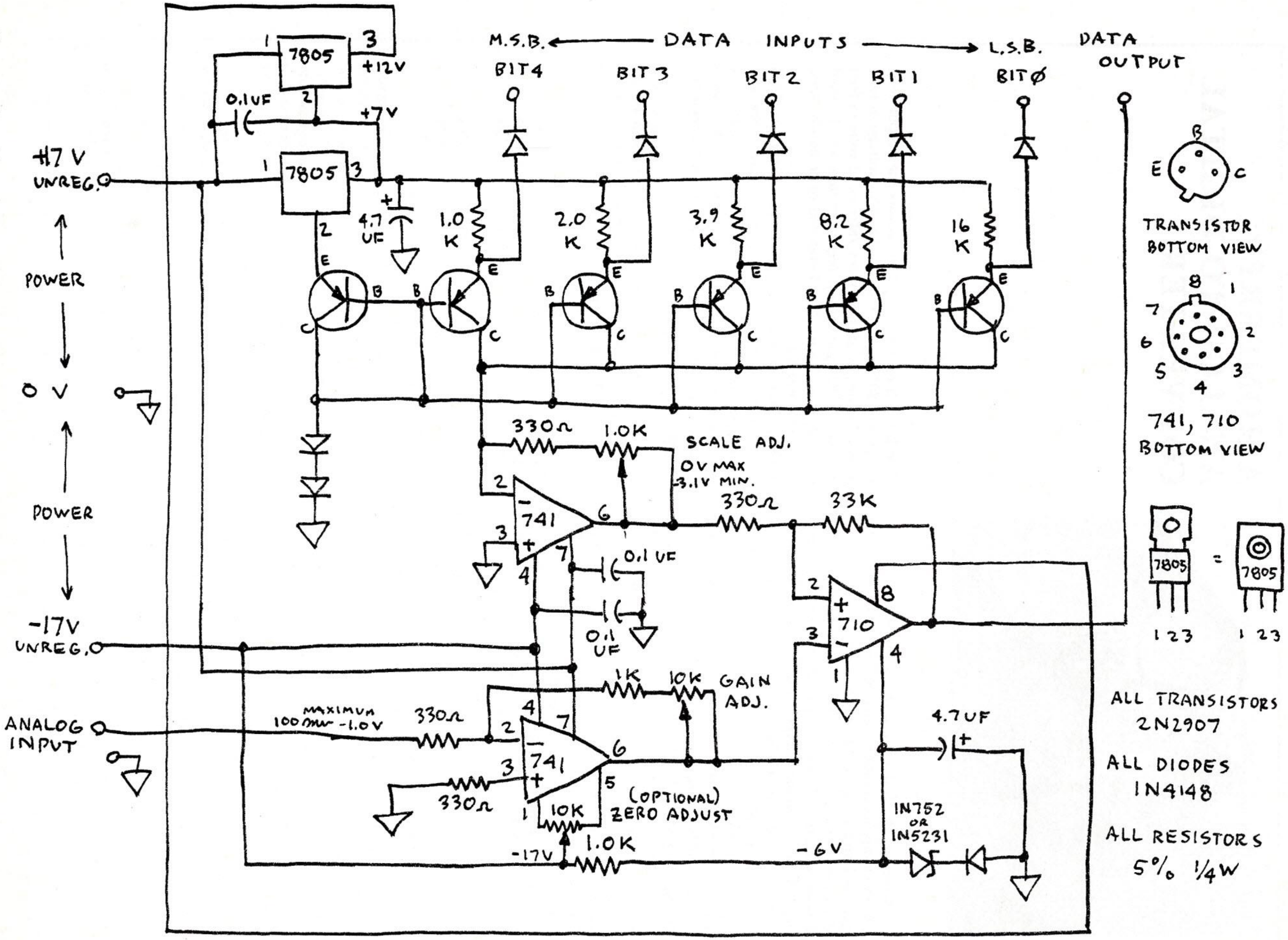
The performance of the converter showed good results. The following table shows the D-to-A behavior (the measurements are taken with a digital voltmeter accurate to 0.35 percent).

| Count | Expected Voltage | Measured Voltage | Error |
|---|---|---|---|
| 31 | 3.10 | 3.11 | .01 v |
| 15 | 1.50 | 1.45 | .05 v |
| 7 | 0.700 | 0.693 | .007 v |
| 3 | 0.300 | 0.310 | .01 v |
| 1 | 0.100 | 0.106 | .006 v |
| 0 | 0.0 | 0.001 | .001 v |

The smallest increment is .100 volt, and the errors should be judged as percentages of the smallest increment. The largest error is thus fifty percent of the smallest increment. Any further extension of the D-to-A converter to more bits would be meaningless considering this inaccuracy. Still, one part in thirty-two is three percent accuracy, which is the best that can be expected of standard panel meters.

The circuit can respond to a full-scale change in data in four microseconds. This is a fast enough response to use with some microprocessors without delays for settling. Faster operational amplifiers like the LM318 will allow a faster settling time, but they are more expensive.

The computer interface must provide latched data at TTL levels capable of sinking at least 7 milliamperes. (CMOS will not work.)


M.S.B. ← DATA INPUTS → L.S.B.
BIT4
BIT3
BIT2
BIT1
BIT0
DATA OUTPUT
7805
3
+12V
1
2
0.1UF
+7V
+17V UNREG.
7805
4.7 UF
+
1.0 K
2.0 K
3.9 K
8.2 K
16 K
E
B
C
POWER
0 V
POWER
-17V UNREG.
330Ω
1.0K
SCALE ADJ.
0V MAX 3.1V MIN.
741
330Ω
33K
0.1 UF
0.1 UF
710
8
4
1K
10K
GAIN ADJ.
MAXIMUM 100mv -1.0V
ANALOG INPUT
330Ω
741
330Ω
10K
(OPTIONAL) ZERO ADJUST
-17V
1.0K
-6V
4.7UF
1N752 OR 1N5231
TRANSISTOR BOTTOM VIEW
B
E
C
8
1
7
2
6
3
5
4
741, 710 BOTTOM VIEW
7805 = 7805
1 2 3
1 2 3

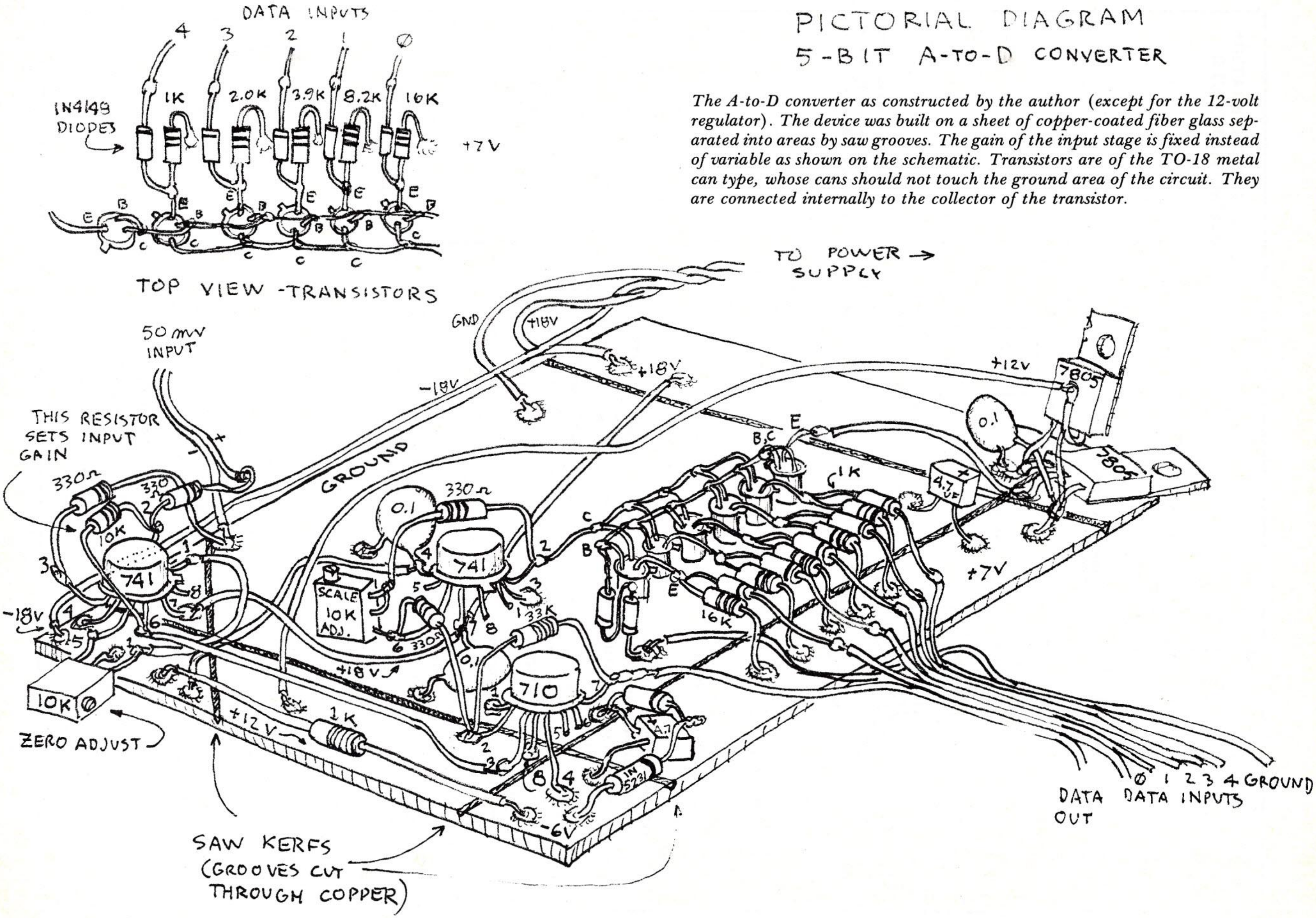
ALL TRANSISTORS 2N2907

ALL DIODES 1N4148

ALL RESISTORS 5% 1/4W

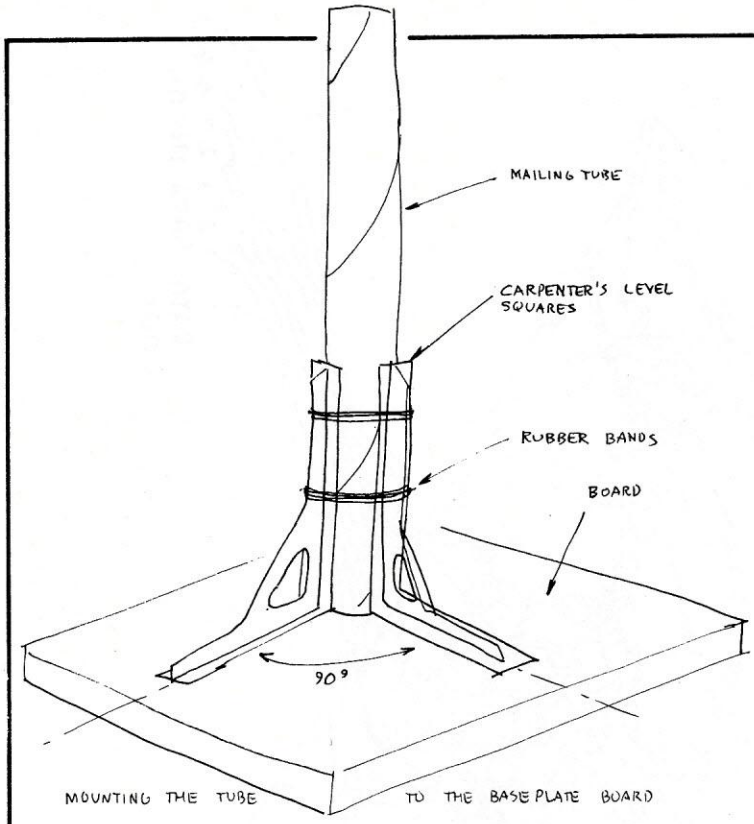
5-BIT D TO A CONVERTER WITH COMPARATOR AND SCALING AMPLIFIER.

L. FELSENSTEIN AUG. 77


PICTORIAL DIAGRAM
5-BIT A-TO-D CONVERTER
DATA INPUTS
4
3
2
1
0
1N4149 DIODES
1K
2.0K
3.9K
8.2K
10K
+7V
E
B
C
TOP VIEW - TRANSISTORS
TO POWER SUPPLY →
GND
+18V
-18V
+12V
7805
0.1
50 mV INPUT
THIS RESISTOR SETS INPUT GAIN
330Ω
330
10K
741
GROUND
SCALE 10K ADJ.
0.1
330Ω
741
33K
330
710
B,C
E
1K
4.7 µF
+7V
16K
10K
ZERO ADJUST
+12V
1K
4.7
1N5231
-6V
SAW KERFS (GROOVES CUT THROUGH COPPER)
DATA OUT
0 1 2 3 4 GROUND
DATA INPUTS

*The A-to-D converter as constructed by the author (except for the 12-volt regulator). The device was built on a sheet of copper-coated fiber glass separated into areas by saw grooves. The gain of the input stage is fixed instead of variable as shown on the schematic. Transistors are of the TO-18 metal can type, whose cans should not touch the ground area of the circuit. They are connected internally to the collector of the transistor.*

# THE LITTLE CALIBRATION SHIELD THAT COULD

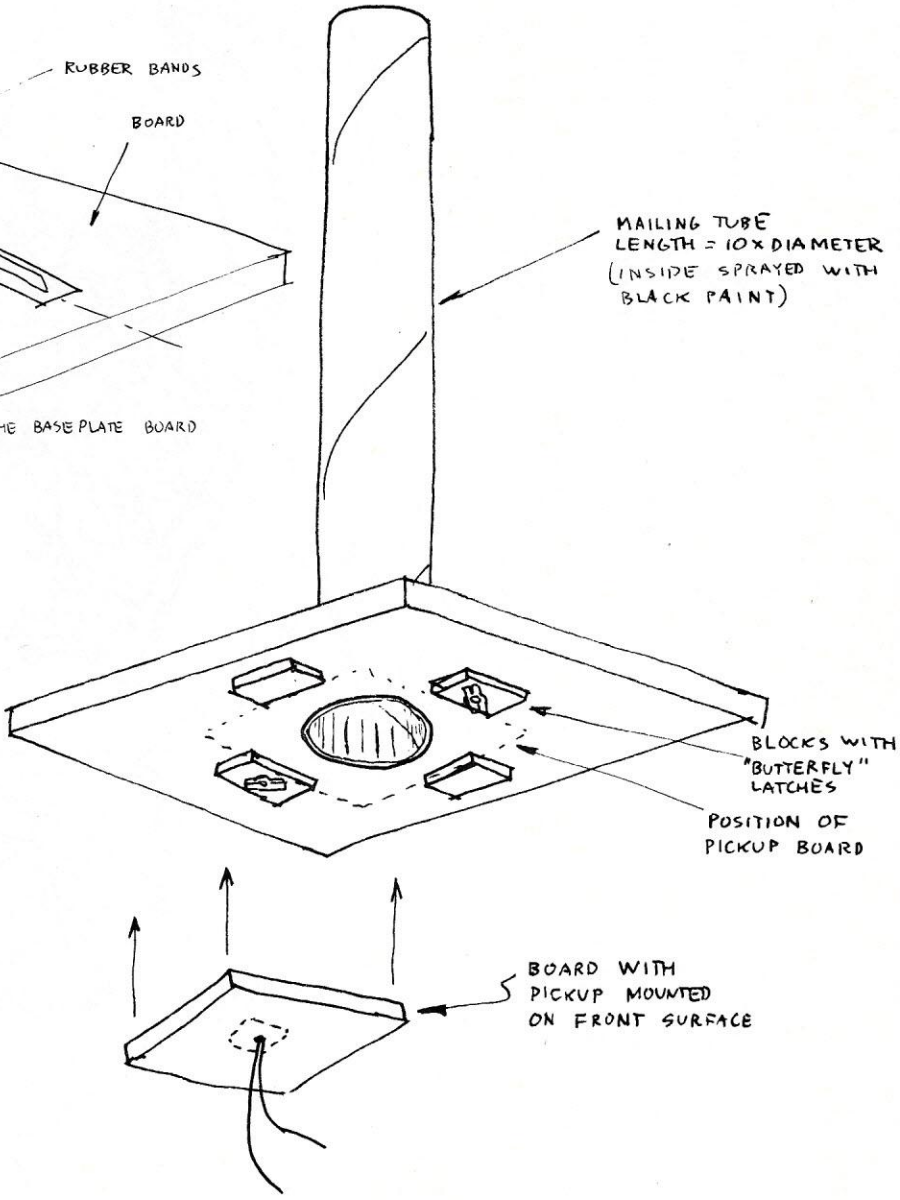

MAILING TUBE
CARPENTER'S LEVEL SQUARES
RUBBER BANDS
BOARD
90°

MOUNTING THE TUBE TO THE BASE PLATE BOARD

DIRECT·NORMAL SHIELD

A direct-normal shield may be fabricated from a mailing tube and a piece of plywood or particle board which is relatively flat. Use a carpenter's straightedge to check for any "bow" or curvature of the board. Ideally, the straightedge should lie flat in contact with the board no matter how it is turned about on its surface. A curvature of one degree will result in approximately 1/16-inch maximum "bowing" at the center of a twelve-inch straightedge.

Cut a hole at the center of the board large enough to allow the mailing tube to be fitted snugly. Using rubber bands, lash the tube to two carpenter's squares, and adjust the position of the tube so that it is perpendicular to the surface of the board (exact term is "normal" to the board). Secure the tube to the board using a quick-setting waterproof glue. Avoid any running of glue through the hole onto the back surface of the board. Similarly, do not allow the tube to project through the board. You will need a clean, smooth surface around the hole on the back of the board.


MAILING TUBE
LENGTH = 10 x DIAMETER
(INSIDE SPRAYED WITH
BLACK PAINT)
BLOCKS WITH
"BUTTERFLY"
LATCHES
POSITION OF
PICKUP BOARD
BOARD WITH
PICKUP MOUNTED
ON FRONT SURFACE

The board to which your pickup is attached may then be clamped to the rear of the direct normal shield when you wish to make a calibration. Simple "butterfly" rotating or swinging latches may be used to secure the pickup, but be careful to avoid placing heavy stresses on the shield board.

check should be done before, as well as after, a run of insolation measurements. Measurements for calibration testing should be "direct normal," that is, taken with the pickup oriented perpendicular to the sun's rays and with all "diffuse" radiation from the sky shielded out. Solar noon—the time halfway between sunrise and sunset (consult you local paper for these times)—is the best time for calibration readings, provided that the sky is clear, free of haze, fog, smog, and, of course, provided that nothing is passing in front of the sun.

Shielding against diffuse radiation is typically done with a tube having a length-to-diameter ratio of ten to one.

To line up the shield, point it toward the sun and sight through the tube until the sun appears centered at the end of the tube. Without wasting much time, position the pickup at the other end of the tube and take a short series of readings (less than five minutes in duration).

Any change in the calibration of your pickup between readings should be added to your data as a correction factor for instrument drift. Your readings can be compared with those of the National Weather Service for your area on the same dates, though your original data should be preserved in case of later indications that the official data were at fault.

You can call your local National Weather Service office and request the relevant insolation data. They should be glad to tell you or to advise you as to where you can obtain published information. They might be encouraged to publish the solar readings along with the other figures released to the newspapers.

> ***Your local National Weather Service office might be encouraged to publish solar readings along with other figures released to the newspapers.***

The official National Weather Service reading divided by your calibration reading will give a calibration factor which is valid for measurements made in the immediate time vicinity of the calibration. However, since you must expect your measurement equipment to change calibration as a result of aging and deterioration of the photocell and other components, calibration measurements should be repeated at regular intervals of a month or so. If you consistently obtain the same factor, you can relax a bit and not calibrate so often. Of course, a sudden change in the factor may indicate some trouble.

Keep a record of the dates on which calibrations are performed along with the readings and calibration factors obtained. This record should be either attached to the pickup or traceable to it by means of a prominent identification number or name written on the pickup.

To assess the solar energy available at a given site, the minimum data to be taken consists of insolation and time of measurement. Fifteen-minute intervals are good for tabulating the data, but each fifteen-minute interval readout should indicate the *average* insolation available during that period. The computer must therefore take data at intervals of about fifteen seconds and compute a running average to be stored in the data array. Additional useful data should start with outside temperature and progress to wind velocity and perhaps wet-bulb temperature to allow calculation of relative humidity. This kind of information is secondary and is generally available from local weather records, but it will be required later on if microclimatic measurement is to be organized to cover a wider climatological range.

For insolation measurements, the pyranometer pickup should be installed at the projected location of the solar panel array and tilted at the same angle of inclination as the projected panels. If the angle of inclination is not preset by a roof slope, it should be set at the latitude angle plus ten degrees.

This may not be the ideal angle for your location, since a reflective ground surface in front of your panels will improve pickup at a higher angle of inclination. So, if possible, several pickups should be set up, each at a different angle, to test for such potential advantages. Or several runs could be made at different angles and orientations. These runs should be of a few days' duration at each angle and should cover, in all, a time span of

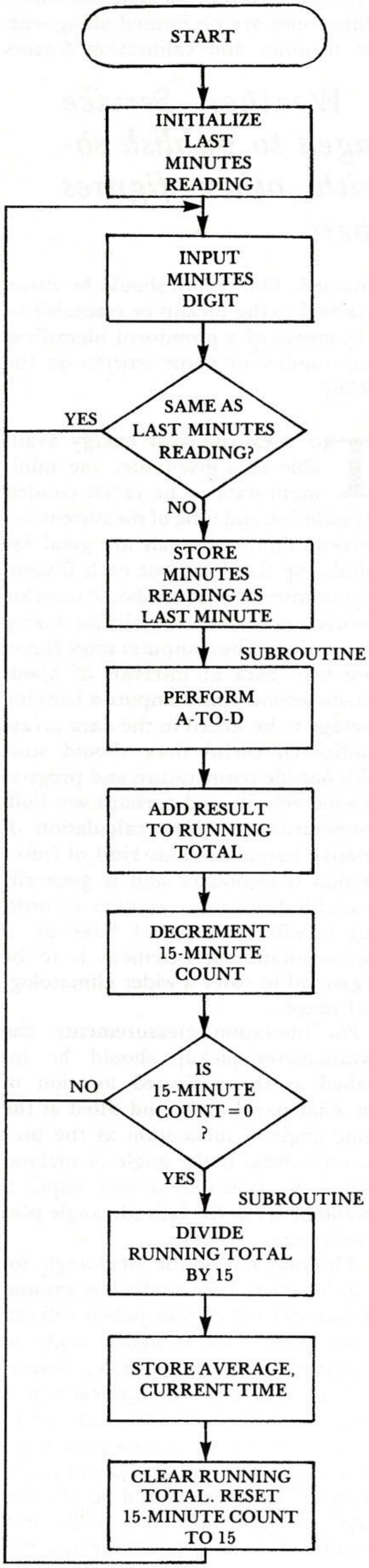
## SOLAR SAVVY

The subject of solar energy has earned a developing literature all its own. Some recommended books are noted below, but they are merely advisory and should not be considered to exclude others. You can't read too many books.

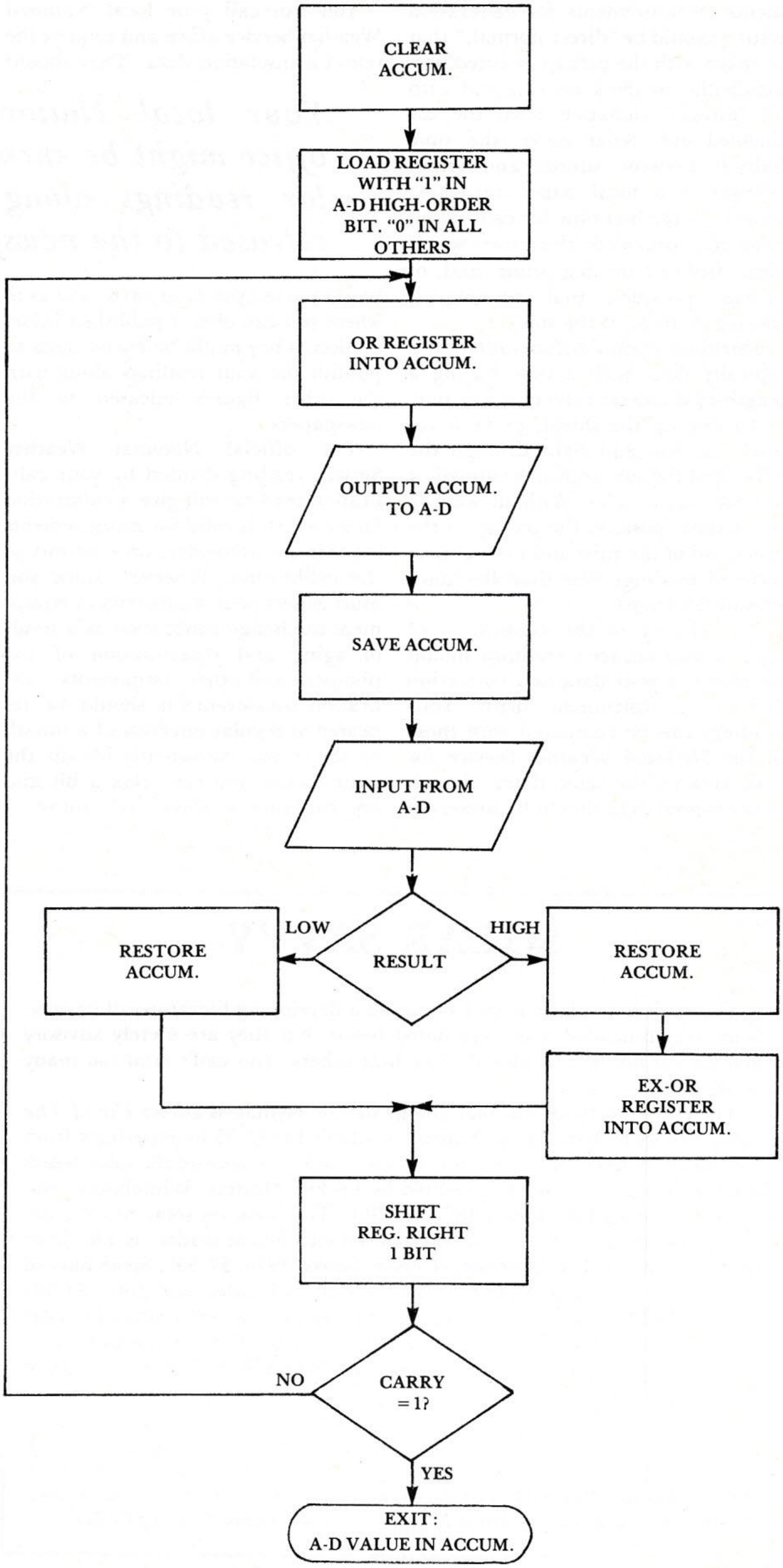
The classic textbook on solar energy for the layman is *Direct Use of The Sun's Energy* by Farrington Daniels (available for $1.95 in paperback from Ballantine Books). A less technical book which can provide the solar basics is *Other Homes and Garbage* edited by Leckie, Masters, Whitehouse, and Young (Sierra Club Books, 1975, $9.95). The book on solar heating described as the "best all-around book for a nontechnical reader" is *The Solar Home Book* by Bruce Anderson (Chesire Books, 1976, $7.50). Steve Baer of Zomeworks in Albuquerque has published a book titled *Sunspots* ($4.50) which is worth reading. Baer is a pioneer in simple low-cost systems for solar heating. The *1977 Sun Catalog* is available for $2.00 from the publisher, Solar Usage Now, Inc., Box 306, Bascom, Ohio 44809. It is a catalog of numerous items, mostly solar, with extensive explanations. A solar meter is included as well as an insolometer from Inservco.

In addition to the general background books on the subject, there is available a series of regional handbooks on solar heating. Published by ERDA (Energy Research and Development Administration), these area handbooks can be ordered through the U.S. Government Printing Office.

DATA ACQUISITION
AND PROCESSING
FLOWCHART


START
INITIALIZE LAST MINUTES READING
INPUT MINUTES DIGIT
SAME AS LAST MINUTES READING?
YES
NO
STORE MINUTES READING AS LAST MINUTE
SUBROUTINE
PERFORM A-TO-D
ADD RESULT TO RUNNING TOTAL
DECREMENT 15-MINUTE COUNT
IS 15-MINUTE COUNT = 0 ?
NO
YES
SUBROUTINE
DIVIDE RUNNING TOTAL BY 15
STORE AVERAGE, CURRENT TIME
CLEAR RUNNING TOTAL. RESET 15-MINUTE COUNT TO 15

SUCCESSIVE APPROXIMATION A-TO-D
CONVERSION FLOWCHART


CLEAR ACCUM.
LOAD REGISTER WITH "1" IN A-D HIGH-ORDER BIT. "0" IN ALL OTHERS
OR REGISTER INTO ACCUM.
OUTPUT ACCUM. TO A-D
SAVE ACCUM.
INPUT FROM A-D
RESULT
LOW
HIGH
RESTORE ACCUM.
RESTORE ACCUM.
EX-OR REGISTER INTO ACCUM.
SHIFT REG. RIGHT 1 BIT
CARRY = 1?
NO
YES
EXIT: A-D VALUE IN ACCUM.

several weeks, so that seasonal changes will be dispersed among all the readings. Obviously, if measurements are made at such different inclinations, you will have to record the angle and other positional information along with the data at each inclination.

Now for the actual programming. By means of a loop, the clock is read and the decision as to when a new minute has started is made. The minutes digit only is read and a comparison with the last-read minute is made. An unequal condition means that a change has occurred. The computer then performs an A-to-D conversion, which should be done in a subroutine, and returns to the main program with the analog value. This value is then added to a pair of memory locations which contain the "running total" of readings within the current. Take care to handle the carry from the low byte to the high byte if your system is of eight-bit word length. Some microprocessors can add register pairs (the DAA instruction on the

**ROMtutorial ROMtutorial**

*Loops:* Segments of a program which are executed a number of times. Usually they consist of a series of instructions followed by a test to see whether the program should execute the preceding instructions, or continue with the instructions that follow.

# THE A-TO-D GUESSING GAME

Analog-to-digital conversion by successive approximation is best seen as a guessing game. The analog value has to be between two limits, let's say, for example, between 0 and 255. The computer guesses a number; the D-to-A converter outputs that voltage level to the comparator, which then returns a single bit response to the computer telling it whether that guess was too high or too low. (Most comparators do not have the capability to tell that two levels are exactly equal.) The computer then tries another guess. This goes on until the computer "narrows in" on the analog value, bracketing it with two guesses which differ by the smallest amount possible (one bit).

There are lots of strategies to use in this guessing game, but on the assumption that you don't know anything about the analog value other than its possible limits, the quickest and surest approach is a "split-the-difference" attack. The first guess is always halfway between the upper and lower limits. In the example given, it would be 128. If the comparator says that the first guess was too high, the computer changes its upper limit to the last value guessed and guesses halfway between the new limits, in this case 64.

If the comparator comes back with a "too low" response, the computer then sets 64 as its new lower limit and guesses halfway between 128 and 64, or 96. The same procedure is followed in successive steps until finally the difference is only one. At that point, if the computer receives a "too high" response, it goes back to the last value, and if the response is "too low," it keeps the new value. The approximation is accurate to within one.

Succesive approximation translates into programming very simply. The range of the D-to-A converter is always equivalent to the range of the binary number represented by the number of bits fed to the converter. Thus, 256 separate levels can be obtained from a D-to-A converter which receives eight bits of data from the computer. Half of such a number is expressed in binary by the high-order bit set to one and all low-order bits set to zero (1000000). 64 is represented by the next-to-highest bit set to one and all others to zero (01000000). As long as the computer keeps getting back a "too high" response, it keeps returning the trial bit to zero and setting the next lowest bit to one.

When the comparator returns a "too low" response, the computer leaves the trial bit set to one and tries setting the next lowest bit to one. The number fed to the D-to-A then increases by half of its previous step. In the original example, 128 was too high and 64 was too low, so the next trial value was 96—in other words, 64 plus 32, or 0110000 in binary.

Suppose a given analog value is a little over 81. Let's examine the guessing behavior of this type of "successive approximation A-to-D converter" by means of a table.

| Trial No. | Binary Value | Decimal Value | Response |
|---|---|---|---|
| 1 | 10000000 | 128 | high |
| 2 | 01000000 | 64 | low |
| 3 | 01100000 | 96 | high |
| 4 | 01010000 | 80 | low |
| 5 | 01011000 | 88 | high |
| 6 | 01010100 | 84 | high |
| 7 | 01010010 | 82 | high |
| 8 | 01010001 | 81 | low |
| done | | | |

As you examine this table, you see that a "one" bit is being shifted from most significant to least significant position, leaving a "one" behind when the response is "low." The conversion is done when the bit emerges from the right end of the word.

A program which performs this sequence is illustrated in the flowchart shown here. A register other than the accumulator is used to maintain a "pointer" consisting of a "one" at the trial bit location and "zeros" in all other bits. This pointer is moved into the accumulator for a trial by using an OR operation. This will not disturb other bits in the accumulator. If a "too high" response results, the program removes the one at the trial location by performing an Exclusive-OR of the accumulator and the trial bit. This will invert the bit at the trial location and will leave other bits unaltered. The Exclusive-OR function is not performed when a "too low" response is received. A bit is left behind in the trial location; the trial bit is then shifted right one position and the process repeats. The conversion is ended when the trial bit appears in the carry or when the trial register is all zero.

8080, for example); this simplifies the problem.

A register or memory location, originally set to fifteen, is now decremented and the result is examined. If it is not zero, the program returns to wait for another change of minute. If it is zero, the program divides the running total by fifteen to extract the average, stores this average and the time in a sequence of memory locations (stack PUSHes become very helpful here), and finally clears the running total and resets the minute count to fifteen.

There is a lot of room for elaboration. You can have the computer read

**ROMtutorial ROMtutorial**

*Subroutine:* A little program nestled inside a bigger program. The computer can "jump to" the subroutine program and begin carrying out its instructions, while marking its place so that it can "return" to where it was after the subroutine is completed. In this way one subroutine can be used by the computer under a wide range of circumstances, without having to rewrite that routine at every point where it would be used in the program.

*Memory location:* An address in the memory of a computer. Like a post box, some place where information can be tagged for later use and kept around unchanged until needed.

*Carry:* An extra bit inside the arithmetic section of processor which indicates that a number has grown too big (after an addition, for instance) and that something should be done.

*Low byte:* In the case where a number is represented by several bytes, like a multiple digit decimal number, this is the smallest of the digits, like the "one" digit in decimal.

*High byte:* In the case where a number is represented by several bytes, like a multiple digit decimal number, this is the largest type of value byte, like the left-hand decimal digit.

*Register pairs:* Temporary storage areas for information inside the microprocessor. Since eight-bit numbers restrict the range of values to 256, it is often necessary to have pairs of registers store and handle the total of sixteen bits between them, allowing values of over 65,000.

*Stack PUSHes:* Instructions in a microprocessor which use a "stack" system for one type of storage. The stack is an area of memory in the computer in which the microprocessor can store information using "last-in, first-out" bookkeeping. PUSH instructions enter a byte into the stack and update the bookkeeping (the "stack pointer").

POP instructions retrieve the last item PUSHed in and alter the stack pointer so that the next POP retrieves the previous item PUSHed.

*Motor control relay:* A relay is an electrically operated switch, in this case, one which can be operated by electrical signals inside a computer. A motor control relay is used to switch on and off the motor of a tape recorder so that it will start and stop the tape when the computer commands.

# SOLAR HEATING

The $64 question in solar heating is, "How big must my collector be to handle a given percentage of the heating of my structure?" If you had ten years' worth of insolation data taken at your site, along with weather data, you would be in a good position to calculate the answer with some certainty. Since most people want results a little faster than that, some approximations are necessary. If you are taking your own insolation data, the approximations are much better than they would be if you were relying solely on tables of insolation.

The following method of calculating "solar heating fraction" has been abstracted from a paper by J. Douglas Balcomb and James C. Hedstrom of the Los Alamos Scientific Laboratory (Los Alamos, NM 87545), where a lot of work on simulation and measurement of solar energy availability is underway. The paper, entitled "Solar Collector Area Required for Space Heating in New Mexico," is being published in *Analysis of The New Mexico Solar Resource* (University of New Mexico, Raymond J. Bahm, editor) by the Energy Resources Board of the State of New Mexico. A draft version was available at the time of this writing, but the book should be out by the time you read this.

The calculation requires some preparation. First, you should calculate the thermal load of the building in BTU/degree-day. The paper explains: "This is the total heat required by the building per day for each one degree F. difference between the inside and the outside temperature. For a small building the major contributors to building thermal load are the heat lost by conduction through the building shell, and the heat used to warm up cold air that has leaked in. Methods of calculating heat losses are to be found in handbooks of the American Society of Heating, Refrigeration and Air-Conditioning Engineers (ASHRAE). For a small, single-story, well-insulated building, the load should usually be in the range of eight to twelve BTU/degree-day for each square foot of building.

"For an existing building, the exact load can be determined from past monthly and annual heating bills and degree-day values. Fuel consumption corrections should be made for furnace efficiency (typically 0.6) and for non-space-heating energy uses. The latter can be determined from summer fuel bills.

"For a proposed building, the thermal load is calculated by adding up the area of each type of external building fabric (walls, windows, doors, ceilings, floors, etc.) times the appropriate thermal conduction coefficient (U-value, in BTU/sq. ft./degree F./hr.), and adding this to the infiltration load as determined by multiplying the building volume times the number of air changes per hour, times the heat capacity of the air (0.018 BTU/cu. ft./deg. F.). This will then yield the building load in BTU/deg. F./hr. Multiply this number by twenty-four to obtain the load in BTU/degree-day which is the number needed in the subsequent step."

Additional information required by the method in the paper is heating degree-days for each month of the year. "One degree-day is counted for each degree F. that the daily median temperature is below 65 degrees F." This data for your area is frequently available from the Weather Service.

Then there is the solar radiation onto the proposed collector. The paper gives a method of estimating this, but you should have much more pertinent data from your data-taking.

the results out to cassette tape, say every four hours or so, if you have a motor-control relay as part of your cassette interface. You can have the program calculate and store the variance and standard deviation of the fifteen-minute groups of readings. You can record outside temperature and wind velocity. Recording this information along with on-time of the house heating system is one good way of estimating heat loss, a necessary quantity to know in heating system design.

You can also enter the sunrise and sunset times for each day or each week and have the machine idle until the sun is up. No need to get up at the crack of dawn just to turn a computer on!

If at this point there seems to be little glamour about the routine taking of measurements, please remember that it is on such measurements that the most advanced and elaborate scientific theories are constructed—and on the basis of which some such theories meet their downfall. Small-scale computers give us the capability to undertake the process of assessment individually, in a practical way, to meet our own needs at the particular sites of our own choosing. Building upon individual experience, participants in Project Prometheus may later choose to combine in group efforts to make valid climatological measurements capable of wider-ranging solar benefit. Fishing for numbers has much more than an abstract technical aspect, since it usually leads to the point where one person says to another, "I've just thought of something really interesting we could make this do!" ▼

## IN FRACTIONS

The paper then defines the Solar Load Ratio (SLR) for each month. "This is the ratio of the total solar energy falling onto the collectors to the total energy required to heat the building; it is given by the following formula:

$$\mathrm{SLR} = \frac{\text{Solar Collector Area (sq. ft.)}}{\text{Building thermal load (BTU/degree-day)}} \times \frac{\text{Total radiation on tilted surface (BTU/sq. ft.)}}{\text{Heating degree-days per month}}$$

Now the paper takes into account the variability of the weather by modifying the SLR into a new number:

$$X = 1.06 - 1.366\ \exp(-.55\ \mathrm{SLR}) + 0.306\ \exp(-1.05\ \mathrm{SLR}) \quad \text{for SLR less than } 5.66$$

$$X = 1 \quad \text{for SLR greater than or equal to } 5.66.$$

The formula for X was derived from a comparison of computer simulations of solar heating systems in twenty-five different locations using five different collector sizes. The formula was created to fit the curve resulting from these simulations. It says that it is safe only to assume 100 percent solar heating when you have 566 percent solar energy available on the average each month. This allows for long cloudy periods and all those other climatic bugaboos which are used to taunt solar partisans.

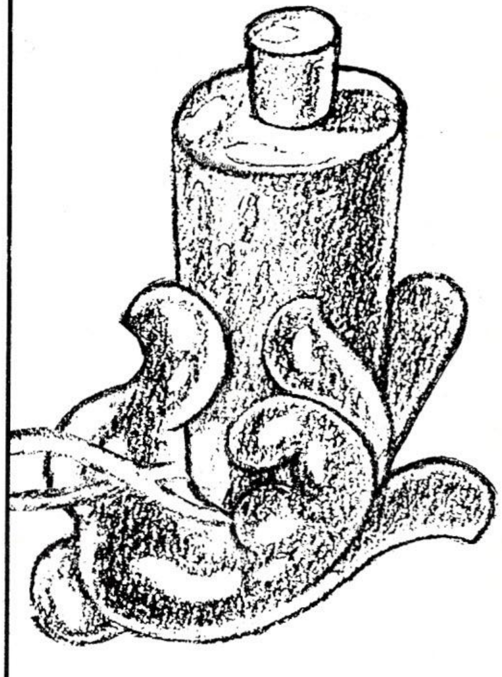
From X the paper recommends calculating the Annual Solar Heating Fraction from the following formula:

$$\mathrm{SHF} = \frac{\text{sum of ([degree-days for each month]} \times \text{[X for that month])}}{\text{sum of degree-days for each month}}$$

where the sums are carried out over the entire year (twelve months). This number is about the closest you can come to a figure on how practical it is to heat your home with a particular collector size and how much auxiliary heat you will need from other sources over a year. A banker should like to have this number when considering financing a solar-heated project.

It is very difficult to go backwards through an equation involving exponentials (exp). Thus, you will have to repeat the calculations of X for various assumed collector sizes if the final size is not determined. It should be a small problem for a computer running BASIC to carry out this calculation and print tables of the results for various assumed collector sizes.

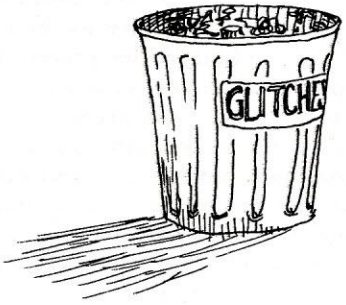
If the banker is still shaking his head, the paper points out that "the average error resulting from the simplified method is essentially zero, and the root-mean-square error (standard deviation) is 4.4 percent solar heating." To be fair, you should point out that the error is between your data and the data resulting from the Los Alamos computer simulations, not observed performance.



*Acknowledgments:*

*Thanks are due to Ray Bahm of the University of New Mexico, Bob Edgerton of Solarex, and Kevin Moran of Edmund Scientific for advice and interpretation necessary to the preparation of this article.*

# BASIC from the Word



Every computer manufacturer has a big book telling you how to program in the version of BASIC that he supplies. Typically, the book is very complete—with detailed descriptions of BASIC syntax and also diagrams depicting data flow to and from the optional floppy disk you didn't buy. If you can get through the whole thing, you can pick up an excellent knowledge of BASIC, down to the last comma and space. But chances are, the one thing the book doesn't teach is how to program fluently.

Or it may be that the book does stress "how to program." If that's so, it probably starts you off with a few "baby programs" which are narrow enough in scope for you to understand with no prior knowledge of BASIC. Then the book probably gradually increases the scope of the examples, adding in the various features of the language so that when you get to the end, you know it all. If you get to the end. If you're at all human, I'll bet you tried to program after you got a third of the way through, right after they introduced the PRINT statement. And it probably didn't work too well.

Those of you who haven't yet struggled through that jungle might like to start another way—by learning how to program first, and then learning how to program in BASIC. BASIC isn't difficult; you can learn the whole language this afternoon. Once you've done that, you can spend the time you would have spent reading the big book practicing instead.

There are, of course, just a few things you can do in BASIC. By stringing these things together in the right order, you can achieve that elusive goal, the bug-free program. Eventually you can, anyway.

In the briefest possible terms, this is what BASIC can do:

BASIC can compute the result of an arithmetic expression, and store the result somewhere. This is called an assignment statement, for reasons I will explain later.

BASIC can print out anything you tell it to—either words of your choosing or numbers that it has computed.

BASIC can accept input from the computer's keyboard (or from your time-sharing terminal, if that's what you're using).

BASIC can test numbers and characters to see if they're equal to, greater than, or less than the value of other numbers or characters.

BASIC can call upon other programs to do some computing and have them return the results of that computation to the place from which it called them.

BASIC can decide (under your supervision, of course) what part of your program to execute, and perhaps how many times to execute it.

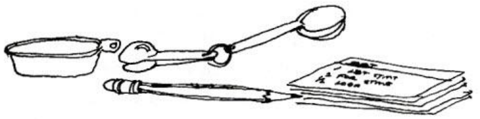
With these six basic BASIC operations, you can program your computer to do almost anything the manufacturer claims it will do. Or you could, if I had told you how to express these things in BASIC. At this point, however, you should try to program without knowing the language because you'll be able to piece your programs together with a minimum of references to the language specifications.

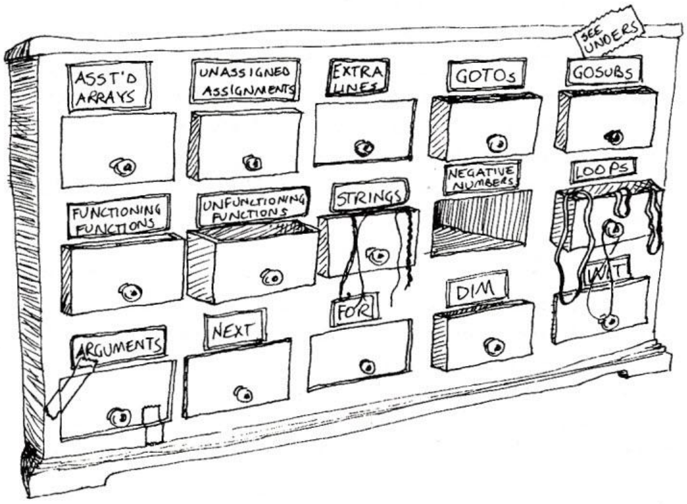
Take, for example, a very simple program to compute the average of a list of numbers. Without knowing a bit of BASIC, you can still select the operations that you'll need to compute this figure. Recalling grade-school arithmetic (you *do* recall grade-school arithmetic, don't you?), remember that the average is computed by adding all the numbers in the list, and dividing by the number of numbers. Can you do that in BASIC?

It would be tempting to say that the first type of BASIC operation, the assignment statement, could handle the whole thing since the process really is just an arithmetic expression. And you could, if you wanted to enter the numbers as part of the program, have BASIC compile it and then run it every time you needed a new average. But I want you to write a program (without knowing a bit of BASIC) that will use any numbers you happen to have and figure out the average—you shouldn't even have to know how many numbers you entered. Because of this nasty re-

# GOTO

### by Eben Ostby





striction (you know, it's for your own good), you can't enter the whole thing as an arithmetic expression; for that you'd have to know how many numbers you were averaging.

The first thing we considered was how many numbers were to be entered. Since I considered it, let us have you consider it. Suppose you wanted to enter seven numbers and average them. You might have a dozen places to put numbers; you could add up the numbers that were input and figure out the total. But if you wanted to input eight hundred numbers, you certainly wouldn't want to store all of them and then add them up. Rather, you'd probably like to add them up as they are input, keeping only the total and a counter of the number of numbers input. When you input the last number, you can divide the total you kept by the counter you kept, and print out the result. If you agree with this analysis, you can try to express it in terms of the BASIC operations I gave you earlier.

This is how I'd approach the program. First, since I'm going to keep a counter, I'd set it to the number of entries input so far, that is, none. This is done with the assignment statement. Specifically, it is done by evaluating the highly complex arithmetic expression "0" and storing the result in the counter. Next, since I'm going to keep a running total, I'd set it to the total accumulated thus far, which is also zero. Now I'd like to accept one number from the keyboard and add it to the running total. This takes two steps: accepting the number from the keyboard and adding it to the running total by using the assignment statement. With the assignment statement, compute the arithmetic expression "total + number input." Then store the result in place of the (old) total.

At this point in the operation look ahead to see what's going to happen at the end of the program. Specifically, to make sure that there will be an end to the program; you don't want to type in BASIC statements until your Blue Cross runs out. Therefore, you have to make sure that if the program averages an arbitrary number of numbers, it will stop after those numbers are input, and if it has, you have the program compute the division and print out the result. This falls under the general catagories of "test a number" and "determine which parts of the program to execute": you test the number input, and if the number is a special, predetermined number, you execute the part of the program which does the division and prints out the result. If the number is not a special, predetermined number, you add one to the current value in the counter and repeat the whole process, starting from where you input the number. That's it; you've just completed your program to compute the average of a list of numbers.

Now that you know how to program in a general sort of way, you can turn your attention to the details of BASIC. There are a few things to keep in mind while you're programming. First, remember what a storage place for a number really is; if you never knew before, I'll tell you. The storage place, which we call a variable, is simply a couple of spaces in the memory of your computer. The reason I tell you this is that your variables may have been part of someone else's program text five minutes ago or part of the BASIC compiler seconds before your program started running.

Most compilers do not bother putting anything in the space that your variable takes up so you had better not assume anything about the value of your variables before you explicitly put something in them. (Some compilers do initialize variables, so if yours does, you can assume that there will be something like a zero in all your variables.) Also, please remember how literally the computer takes everything. Lacking intuition about your intentions, it will seem unforgiving when it is, in reality, blind.

Back to what BASIC can do. We'll first look at the assignment statement. In BASIC, the assignment statement looks like this:

*LET A* = expression

Here *expression* is my shorthand for the guts of the matter, the computational part of the statement. It means that the expression will be evaluated and the result will be stored in the place called A. You can substitute the name of any variable for A; if the variable doesn't exist already, BASIC should create it for you. The rules for what you may call a variable change

Illustrated by the author

from system to system, but many require that it be one or two characters long—the first character an alphabetic character (A-Z) and the second a digit (0-9). Other more elaborate systems let you use any string of letters up to six or eight characters, with the restriction that the first be a letter (alphabetic) and the remainder be either alphabetic or numeric. To keep myself out of that mess, I'll use single letters, and perhaps you should for now, too.

Returning to the expression, which is a computerized version of a simple arithmetic statement, you'll find that BASIC's rules are at least as rigid as those of a grammar-school teacher although they are different. The five basic operations in BASIC are addition, subtraction, multiplication, division, and exponentiation (raising to a power). They are represented by the symbols

+ — */^

respectively. (Again, there are exceptions—some oddball BASIC systems that were written by FORTRAN freaks use a double star ** to represent exponentiation. If yours does, consider yourself unique and make the changes where necessary.) You can do any of these operations with any numbers or variables, for the most part; you can add 2 to X or raise X to the Y power like this:

```
LET N=2+X
LET M=X^Y
```

*A variable is a storage place for a number, a couple of spaces in the memory of your computer.*

You can also put a number of operations into one expression. When you do this, BASIC will do exponentiation first, followed by multiplication and division, and in turn by addition and subtraction. If you want to compute things in a different order, you can use parentheses to force BASIC to do it your way; the operations inside parentheses will be done first so that you can multiply a number by a sum like this:

```
LET S= (A+B) *5
```

If you had written,

```
LET S=A+B*5
```

it would have been the same as saying,

```
LET S=A+ (B*5)
```

which is not what you wanted.

Displaying stuff on your terminal or screen is very easy in BASIC. All you say is PRINT:

```
PRINT "THERE ARE GOBLINS
IN THIS MODEM"
```

will cause BASIC to print out a serious warning. You can also print out numbers that you've computed:

```
PRINT A
```

will cause BASIC to print out the current value stored in A. You can also mix these two options together. Here you have a choice. You can print something—for instance, a variable—and follow it immediately by something else—for instance, some text in quotes:

```
PRINT A;" IS THE RESULT
OF THE CALCULATION."
```

If the value in A is, say, 155, this statement will print,

```
155 IS THE RESULT OF THE
CALCULATION.
```

You can have BASIC skip to the next tab stop (yes, even if there aren't physical tab stops on your machine—BASIC simulates them anyway) by separating the data items by commas instead of semicolons. BASIC, by the way, does an automatic carriage return at the end of every output, but you can over-ride this return by putting a comma or semicolon at the end of the PRINT statement.

In order to get data from the outside world into your program, you've got to instruct BASIC to accept it from the keyboard. Here BASIC only needs to know where to put the data, so the command is relatively simple:

```
INPUT A
```

will ask for input from the keyboard and store what was entered in the variable A. Now you can treat A like it was any number and use it in a calculation or print it out. You'll find that when the INPUT statement is executed, BASIC will print out a question mark, and wait for input. When you finish entering the number, you hit the return key, and BASIC continues on its way with your number in its clutches.

With these first few elements, you can write the simplest of programs, simpler even than a program that averages numbers. For instance, you could write an adding machine program—one that accepts two numbers and then adds them together. If you did, it would probably consist of these elements: a PRINT statement to write a prompting message on the display; an INPUT statement to accept one number; another PRINT statement and an INPUT statement; a LET statement to add the numbers; and, finally, a PRINT statement to output the result of the addition. Now you have to put the statements together in a program. This little formality is a means of telling BASIC how the statements fit together, that is, which comes after which.

You begin by numbering each instruction with a positive integer (no sign, no decimal points allowed). Generally, the largest number you can use is 9,999, and the smallest is 1. Do many programs have 9,999 statements in them? you might ask. No, they don't. You can use such a large number in order to leave room between your statement numbers: most programmers number their statements 10, 20, 30 instead of 1, 2, 3. In this way, if you want to insert a few more lines of program text between two existing lines in your program, you can do so without having to re-enter your entire program from that point on. If you wanted to enter a new line between numbers 140 and 150, you'd simply enter the new line with number 145.

Thus, to enter a program into BASIC, you start each line with a number, and BASIC takes the presence of the number as a cue to store the stuff that follows it as a program line. When it comes to the time to run the program, BASIC will execute the line with the lowest number first—whether or not it was the first line you entered. (Yes, you *could* enter the whole thing in reverse order, if you really wanted to. And to change a line, you just re-enter the line number, and follow it with the new line.)

The line numbers also serve to identify each line. In this way, you can have your program executed in some order other than the normal ascending-line-number order. For instance, you can write a *loop,* a section of code that repeats itself. As an example, why don't you write down your interpretation of the adding machine problem, in BASIC. Mine looks like this:

```
10 PRINT "ENTER FIRST
   NUMBER"
20 INPUT A
30 PRINT " ENTER SECOND
   NUMBER"
40 INPUT B
50 LET C=A+B
60 PRINT "TOTAL IS ";C
```

To run the program, after you've entered it, type RUN, and hit the return key. Now when my particular version executes, it will ask for both numbers, sum them, and print out the result—exactly once. To do it again, you have to type RUN again. If you want the program to go back to ask for input right after it prints the results, you can add another line to the end of the program which tells it where to go (if you will):

```
70 GOTO 10
```

When BASIC encounters a GOTO statement, it goes and executes the statement whose number appears as the argument to the GOTO instead of executing the next sequential instruction. By adding the line number 70, you've told BASIC to start over again—automatically—and to run through the preceding sequence of instructions. Once it repeats the sequence again, what will it do? Well, it will again encounter statement 70, which instructs it to go back and start once more. This process will never stop, at least not under its own control. I've never met a system that didn't have a provision for interrupting this sort of thing, however, so look for yours. It may be a key labeled Attn, Break, Escape, ESC, DEL, Request, or Interrupt, or it may be a specific combination of keys: you may have to hold down CTRL and enter a C or a B or a K, or you may have to enter the word STOP and hit return. On some systems, you have to disconnect the terminal for a second, and on some computers, you may have to restart the system at some point. Usually, though, it's not difficult.

***Remember how literally the computer takes everything. It will seem unforgiving when it is, in reality, blind.***

You can also have BASIC decide whether to execute a GOTO. If you used this option, you could have BASIC decide when to stop looping. To do this, you could use the IF statement:

```
IF expression THEN line
IF expression GOTO line
```

These two forms of the IF statement say the same thing (in fact, they're interchangeable on many BASIC systems although some systems allow only one or the other). In this case, *expression* is not the same as the *expression* in the LET statement. Here the *expression* is something that you can interpret as being either true or false—it must be an equality or an inequality. These are some *expressions:*

```
A=B
A<B
A<=B
A>B
A>=B
A<>B
```

They're read, "A is equal to B," "A is less than B," "A is less than or equal to B," and so on. The last one you could read, "A is less than or greater than B," but what that means is really "A is really not equal to B." So you can insert this sort of expression—substituting any variable or number for A or B—into the IF statement. The *line* is just a line number. When it appears, BASIC evaluates the expression. If it is true—for instance, if the expression reads A>90 and the value in A is 100—then BASIC Goes To the line number you used. For example, in the adding machine program you could decide to have the program stop when a zero total was printed. Since the total is stored in the variable C in the program, you could replace the line numbered 70 by this line:

```
70 IF C< >0 THEN 10
```

and in that way, you would have control over your runaway program once again. Since, if the *expression* is not true, BASIC just proceeds to the next line, you could also write line 70 as a couple of lines, like this:

```
70 IF C=0 THEN 90
80 GOTO 10
90 STOP
```

STOP is another way you can control your program. When a STOP statement is encountered, BASIC simply stops running the program, just as if the program had ended.

Now you know enough to translate that rough sketch of the averaging program into BASIC code. I'll make a suggestion: have the user enter some unusual number, say 99999, as a signal that he's entered all the data for the average. The average program might look like this:

```
10 LET C=0
20 LET T=0
30 PRINT "EACH TIME A ? IS
   PRINTED, ENTER ONE
   NUMBER FOR THE
   AVERAGE. WHEN YOU'VE
   FINISHED, ENTER 99999"
40 INPUT A
50 IF A=99999 THEN 90
60 LET T=T+A
70 LET C=C+1
80 GOTO 40
90 A=T/C
100 PRINT "THE AVERAGE
      OF THE ";C;"NUMBERS
      ENTERED IS ";A
110 STOP
```

If you had known at the beginning of the program how many times the "loop" of statements 40-80 were to be executed, you could have used another BASIC instruction that is made for looping. It's called the FOR statement. Although what it does can be easily done in other ways, the FOR statement simplifies programming (and hence life) for you. It looks like this:

```
FOR A=B TO C STEP D
```

and always following it somewhere is another statement that looks like this:

```
NEXT A
```

In my example, A, B, C, and D can be any numbers or variables. Most—but not all—BASIC systems allow you to use expressions, like those used in LET statements, in place of B, C, and D. Finally, the part "STEP D" can be omitted. This is what it does: when the LET statement is encountered as the program runs, BASIC evaluates B—that is, whatever you used for B—and assigns the result to A, just as in the LET statement.

Now BASIC compares A to B and C. If A's value lies between B and C, BASIC continues executing the program. If A's value is not between B and C, then BASIC finds the statement NEXT A. (Remember that A can be any variable name. If your FOR statement reads FOR X=1 TO 10, then BASIC will look for the next statement which reads NEXT X.) When it finds NEXT X, it goes on and executes the program that lies after the NEXT statement. If, however, A's value was between the values of B and C and the program right after the FOR statement was thus executed, the program will eventually reach that same NEXT statement. When it does, BASIC will add the number D to A, or if you omit the STEP clause, BASIC will add the number 1 to A. Then it will go back to the FOR statement and test A against B and C again.

*The FOR statement simplifies programming (and hence life) for you.*

If you don't quite have that, look at this simple example of a BASIC program:

```
10 FOR I=1 TO 4
20 PRINT "LOOP NUMBER ";I
30 NEXT I
40 PRINT "ALL DONE"
```

When you run this little program, what will happen? It's very important that you learn to figure out what the computer will do, so this sort of exercise is worth doing.

This program will print the following on your terminal:

```
LOOP NUMBER 1
LOOP NUMBER 2
LOOP NUMBER 3
LOOP NUMBER 4
ALL DONE
```

I must warn you that different BASIC systems do different things when you use a negative number for the STEP, or when you start A out not less than C. Depending on what version of BASIC you use, you may or may not get in trouble if you write the statement:

```
FOR I=1 TO 0
```

Ideally, that statement shouldn't execute any loops, but on certain systems it executes as many as 65,000.

The FOR statement makes a good introduction to a very important feature of BASIC, something called an *array*. An array is a list of numbers, all stored in the computer at one time. But it's not like having many different variables to hold all the numbers because the array has one name. To determine which number in the list you want to use, give BASIC an *index* or a *subscript* into the array. First, however, you have to tell BASIC that you're using an array, and how big it is. For this, use the DIMENSION statement:

```
DIM A(20)
```

tells BASIC that you're using an array called A, and that it has twenty elements. When you want to assign anything to a particular element of the array, give the name of the array and the subscript when you write the assignment statement:

```
LET A(1)=Z+2
```

This puts the value obtained by adding Z and two into the first element of A. The subscript can be an expression, too:

```
LET A(X-1)=A(X-2)
```

This statement moves some number from one spot in the array to the next higher spot.

One use for the array could be to store all the numbers that you want to average. When you have a moment, why not rewrite the averaging program to store the numbers in an array as you enter them. Remember that the counter, C, tells you which number is being entered: it's zero before you enter the first number, etc. Once you do this, you can add on a bit of program to make all the numbers you input print out after the average was printed. This is how I'd do it, assuming the array is called N:

```
110 FOR I=1 TO C
120 PRINT N(I)
130 NEXT I
140 STOP
```

Another useful BASIC item is the *string*. If a variable name is to denote a string, you follow it with a dollar-sign, like this:

```
INPUT A$
```

Strings act very much like ordinary variables, except that they can hold letters instead of numbers. Thus, when you execute the above INPUT statement, you can enter a person's name or address or hair color instead of a number. You can then use the string in any other statement; however, you can't add or subtract or multiply or divide (or whatever) a string. You can test it in an IF statement, though, and you'll find that

*"ANDY"*

is less than

*"BETTY"*

which is less than

*"BOB."*

Finally, suppose you have written a program which uses, say, an array. Suppose further that, in two places in the program, the programmer inputs a number that's going to be used as a subscript to the array. If this seems far-fetched, take as an example a program which stores test scores in an array. You might want to update one student's test score; later you might want a report of what a particular score was. Each time you'd have to input the number where the score was stored. If the person entering this number entered one which wasn't in the proper range for a subscript for the array, you'd want to tell him or her, right? This takes a few BASIC instructions.

My point here is that you'd have to do the same thing in two different parts of the same program. You'd be doing the same thing twice; it seems wasteful to repeat the BASIC instructions if there is some way to reuse those instructions. And I wouldn't have mentioned it if it wasn't possible. The way to do this is with a *subroutine,* which is just a section of code to which you can go and then return to the point from whence you came. In BASIC you use a GOSUB statement to execute a subroutine—a GOSUB seems to act just like a GOTO since it just makes the program continue with the line number you gave it. But when you find a RETURN statement, the program continues with the statement after the GOSUB statement. Got that? It looks like this:

```
•
•
•
110 PRINT “BEFORE”;
120 GOSUB 900
130 PRINT “AFTER ”;
140 GOSUB 900
150 PRINT “FINISHED”
•
•
•
900 PRINT “ DURING ”;
910 RETURN
•
•
•
```

When this little bit of code is run, BASIC will print out

```
BEFORE DURING AFTER
DURING FINISHED
```

which may be a cryptic way of explaining the GOSUB statement, but if you can figure it out, you have the essence of it.

There's yet another kind of subroutine in BASIC. This one is easier to use, though. It's called through the LET statement. The idea is this: you want to find the value of some function of a number. Amongst the functions you might find in a BASIC system are logarithms, trig functions, absolute value (the positive version of a negative number), the floor or integer part of a number, and so on. BASIC has many pre-defined functions and you can find out which yours has by looking in the appropriate manual. You use most functions by putting their names into a LET statement, as if they were variables:

$LET\ A = B + SIN(X)$

Here the function is called SIN (which means *sine* rather than anything else). The number of which you want to take the sine inside the parentheses; it is called the argument of the function. In this example, BASIC will compute the sine of X, add B to that and store the result in A. There are usually lots of different functions in BASIC: here are some common ones and what they're for:

| | |
|---|---|
| ABS(X) | absolute value of X (positive number) |
| INT(X) | integer part of X |
| SIN(X) | sine of X |
| COS(X) | cosine of X |
| TAN(X) | tangent of X |
| MOD(X,Y) | remainder when X is divided by Y |
| RND(X) | random number between 1 and X |
| EXP(X) | $e^X$ |
| SQRT(X) | square-root of X |

By this time, you probably have given up reading this wordy article as well as whatever other literature you had on BASIC, and you've probably started programming. My advice to you now is to program as much as you can, using as much of the language as you can. Once you are used to it, you can throw expressions around like mad. It's a good idea, too, to look at other full-scale programs and figure out how they work; you'll pick up a bit of programming style in the process of learning the language. But the best thing I can suggest for now is to have a list of some BASIC statement choices at hand while you program, so you can pick the right one for the right job. ▼

## BASIC *BASIC* STATEMENTS

| | |
|---|---|
| *PRINT “STUFF”; N,* | Prints out stuff within quotes and value of variables. A semicolon allows you to string things together on one line; a comma does the same but skips to the next tab stop. |
| *LET A = B + (C—D/S(2))* | Evaluates the expression on the right of the = and stores it in the place named on the left of the equal-sign. |
| *IF I=J THEN 999*<br>*IF G+3 >=88 GOTO 15* | Figures out if the logical inequality (or equality) is true, and if it is, goes to the line named. |
| *GOTO 50* | Makes the program continue execution with the named line. |
| *GOSUB 9044*<br>*RETURN* | Makes the program continue execution with the named line but sets things up so that a RETURN statement will make execution continue with the line after GOSUB. |
| *FOR A = B TO C STEP D*<br>*NEXT A* | Starts A off at the value B, executes the stuff up to the NEXT statement, increments and tests A to see whether the program should continue or go to after the NEXT statement. In other words, sets up a loop. |
| *DIM V(123)* | Alerts BASIC to the presence of an array called V with 123 elements; V can then be used in any statement. |
| *INPUT A* | Accepts the value for A from the keyboard. |
| *LET A$ = “CHARACTERS”* | Assigns the characters to the string A$. |

# Chipmaker, Chipmaker, How Does Your Crystal Grow?

### by Sandra Faye

What do these things have in common? Diamonds. Table salt. Aluminum. Ice. How about a hint? Rock candy. Right. They are all crystals, solids with a latticework of predictable angles, usually right, and flat surfaces, often squares or rectangles, all repeated indefinitely. Window glass, on the other hand, is amorphous. It may have crystalline spots in it, but most of it is an interwoven mesh of angles and plane faces, perhaps regular, even symmetrical, but not repeated endlessly in a long-range order throughout the whole plate.

Glass is silicon, by and large oxygenated compounds of silicas and silicates mixed with soda and lime. Silicon is the second most abundant element on earth, surpassed only by oxygen. It is to the mineral world what carbon is to the organic world, the building block of most compounds. As silicon dioxide, it forms sand, flint, quartz, and opal. As complex silicates of metals, it participates in almost all rocks, clays, and soils. It is, in fact, a semimetal, naturally a fair insulator and a moderate conductor—a semiconductor. THE semiconductor. Its resistivity, a measure of its capacity to conduct an electrical current through itself, which varies with temperature and structural impurities, like human conception, is no longer left to accidents of nature.

*The growth of silicon, like human conception, is no longer left to accidents of nature.*

Silicon for computer chips is never left to chance. As it grows, technicians control how pure or impure they want the finished silicon to be, and by melting and freezing bulk silicon (the freeze takes place at temperatures above 1400 degrees Celsius), they create a simple giant crystal of silicon, as much as four inches in diameter and thirty inches long, a cylinder the color of moonlight on a Hawaiian reef, silvery and blue, with reflections that seem to radiate from inside the ingot rather than simply bounce off of its mirrored surface. One crystal, almost the size of a small stove pipe, is solid glass, with the radiant tone of the finest lead crystal. One crystal, spun so precisely that its internal skeleton lies

Photographs courtesy of Siltec

*Polycrystalline silicon, the ultra-pure silicon before it is melted down for semiconductor use.*

straight and true throughout its whole body, as predictable and perfect as the small seed around which it grew to rigid specifications. From this crystal will be sliced an average of fifteen hundred teal blue mirror-finished wafers, and on each of these wafers will be printed two hundred and fifty or three hundred microprocessor chips.

Only in the last few years has silicon crystal growing become an automated procedure regulated by an operator who has learned his craft in a two-week course. Before then, pulling crystals was an arcane art, entirely dependent upon the skills of the operator. In Menlo Park, California, Siltec's first product, in 1970, was a crystal growing furnace, capable of turning out uniform three-inch silicon ingots. Now, although it continues to sell its equipment to other manufacturers, Siltec's major product is two-inch, three-inch, and four-inch wafers.

*The melt, and the furnace.*

Siltec buys its bulk silicon from outside sources. These suppliers take clean sand, which is essentially silicon dioxide, and, in one way or another, get rid of the oxygen. The most venerable method burns the sand in chlorine gas so that a gas of hydrogen chloride (in liquid form, it's hydrochloric acid or bleach) goes up an exhaust vent and the residue is a liquid pool of silicon tetrachloride, two atoms of silicon holding four atoms of chlorine. Any remaining oxygen and hydrogen are driven off during distillation of the silicon tetrachloride in a multistage still. To get rid of the chloride the processors might take the distilled silicon into a furnace containing a hydrogen atmosphere (instead of our usual oxygen one), and place a glowing molybdenum tube heater in the center.

In the heat, the silicon comes to rest, atom by atom, on the two-foot moly tube, and the chlorine goes off as a gas into the air, combining with hydrogen in the atmosphere to become our old friend hydrogen chloride gas. When the deposition is complete, a two-inch thick tube of silicon covers the moly tube. One need only melt or dissolve away the moly tube, and break apart the silicon tube into rods and chunks.

This polycrystalline silicon, ultrapure, arrives at Siltec looking like nothing so much as some exotic metal ore, here as shiny as new silver, there as black as cast iron. An operator in the furnace room places seven thousand grams, seven kilograms, almost sixteen pounds, of polycrystal in a crucible made of quartz, another silicon dioxide crystal. The frosty white crucible looks like a streamlined four-quart souffle dish. Pure silicon impedes the flow of electrical current through it, making it a pretty good insulator. This ability is called high resistivity. In most true metals, electrons move from space to space within each level of the atom. Since electrons are considered a negative charge, this type of metal conductor is called a negative or n-type conductor. In a few metals, whole bands or levels are empty of electrons and when the metal is heated, electrons jump over from some other band to fill these holes. These holes are said to be positive and, thus, these metals are p-type conductors. Silicon is naturally a p-type conductor, but unlike metals, this property may be adjusted by adding intentional impurities called "doping agents" or "dopants." If a customer wants to alter the conductance of the silicon, the operators at Siltec mix into seven kilograms of polycrystal about two grams of dopant, any combination of boron, phosphorus and antimony.

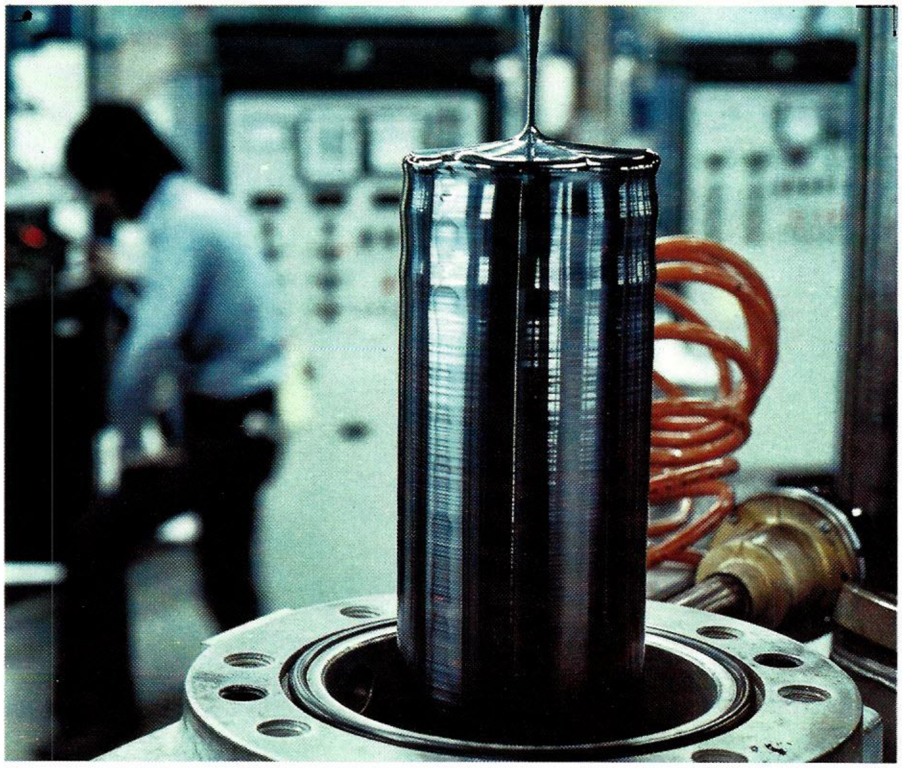
*In temperatures over 1400 degrees Celsius, the silicon comes to rest, atom by atom, on a two-foot moly tube.*

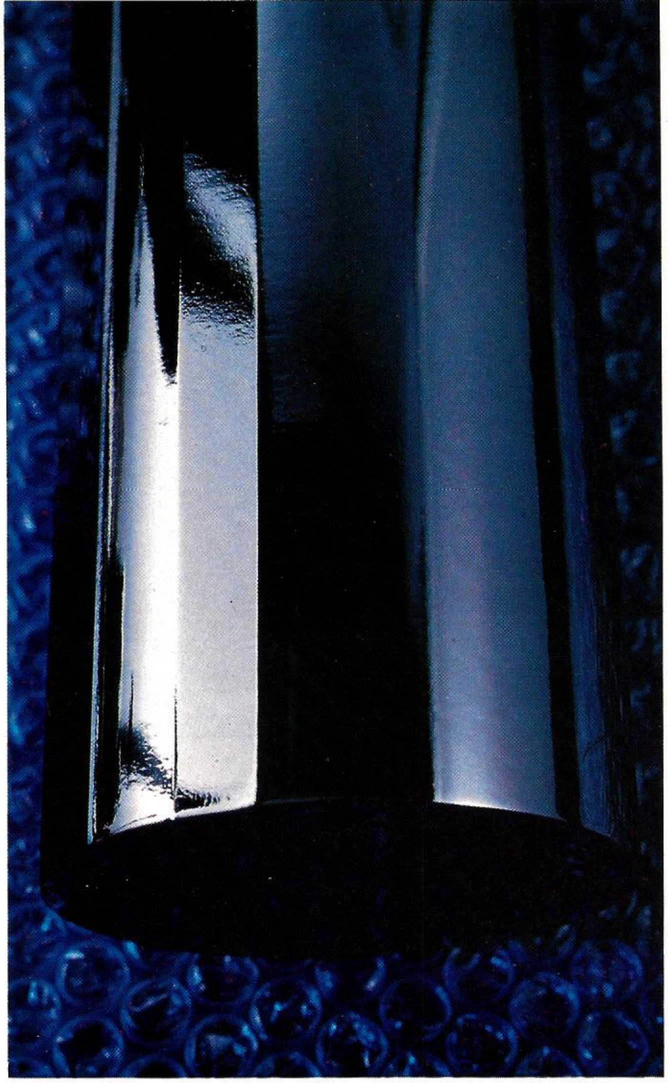
The crystal fits inside a charcoal gray-colored susceptor, a larger souffle dish made of graphite, one of the two major crystalline forms of carbon. (The other is diamond.) The susceptor in its turn slides into a slatted graphite heater, which in turn fits into a larger cylinder made up of three layers of graphite wrapped with graphite felt tied on with moly wire. The whole contraption fits inside a special furnace designed by Siltec's president, Robert Lorenzini. A high current of electricity is sent through the heating element around the susceptor, to heat the crucible as if it were a pot on a stove.

Over the crucible, a six-inch long seed crystal, cut from the quarter-inch diameter master crystal rod, dangles out on the edge of a needle-shaped assemblage. The seed is connected to the seed chuck; the seed chuck is connected to the seed rod.

There are two sorts of template seeds in semiconductor silicon crystals: 1—1—1 has four rays, in the shape of a cross across the top of the crystal. These rays extend vertically through the body of the ingot like four symmetrical vertebral columns; 1—0—0 has six rays, like an asterisk, these rays also cut vertically through the body. The customer orders the orientation. 1—1—1 is preferred for building bipolar devices, 1—0—0 for metal oxide semiconductors. "That's just what works

*The finished ingot coming out of the furnace. From this crystal will be sliced an average of fifteen hundred teal blue mirror-finished wafers, and on each wafer will be printed two hundred and fifty or three hundred microprocessor chips.*





*An ingot, showing primary and secondary flats. The flats identify the internal structure of the crystal and of each of its wafers.*

*A row of polishing machines, in which wafers are buffed to a mirror finish on one side.*

better. There isn't always an important theory to explain things, you know."

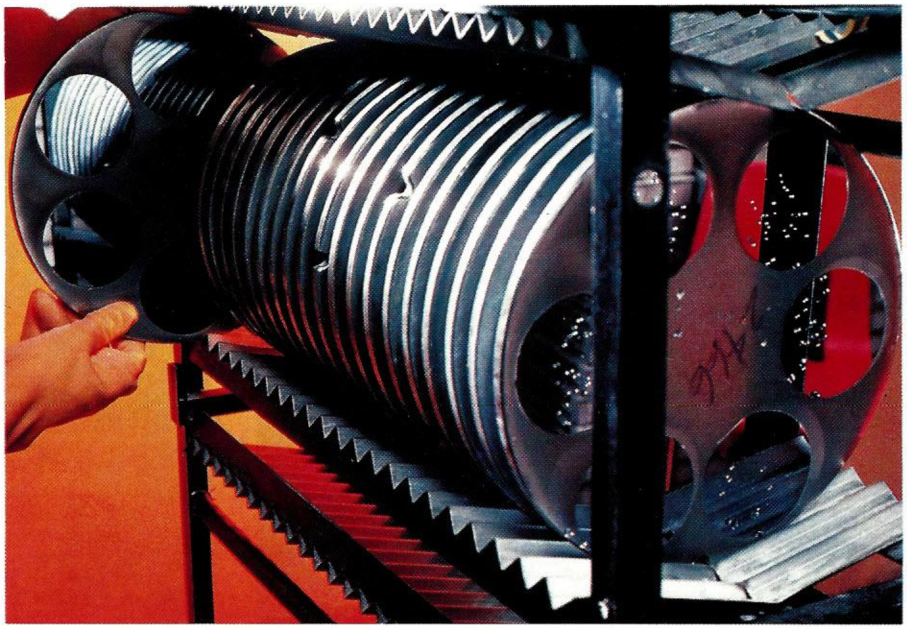
The seed rod hangs from a coil-shaped mechanism that winds and pulls the seed up out of the melt as the crystal grows around it. Assembled, the furnace and coil look like an armored apple cider press equipped with view ports for sightseeing.

*The furnace and coil look like an armored cider press.*

When the silicon polycrystal reaches 1420 degrees Celsius, it melts. From then on the crystal grows by freezing to the seed, revolving in its midst, at about 1418 degrees Celsius. A little heat travels up the seed rod and is lost from the melt. That plus some very fine automatic temperature adjustments by the microprocessor controlling each furnace keeps all the silicon molten except that just touching the seed. The crucible rotates in one direction, the seed on its rod rotates in the other. As the silicon freezes around the seed, forming itself into a larger and larger version of the seed itself, the operator pulls the seed out of the melt, quickly for the first twenty to forty minutes, to form a four- or five-inch thin neck, just a bit wider than the seed, and then slower for the next fifteen to forty-five minutes so that the crystal widens out into a shoulder of the desired diameter of two, three, or four inches. Once the shoulder has formed, the microprocessor takes over, with a little further help from the operator, and pulls the silicon ingot some thirty inches down from the shoulder.

The operator takes over again for the last couple of hours to tail off the crystal, bringing it to a point so that it can be broken from the small puddle of residue once everything hardens, after the furnace has been turned off. A thin umbilicus runs from the tip of the tail to the puddle of remaining silicon. On the 1—0—0 crystal the latticework of rays or nodes stands out in relief on the shoulder and down the sides of the ingot. On the 1—1—1, the nodes don't show up on the shoulder, but form flat stripes along the ridged sides of the ingot.

Once the furnace has been shut down, it takes about two or three hours for the ingot to cool down enough to be handled with asbestos gloves, and another two and a half or three hours before it approaches room temperature. Something there is that abhors a



*Superflat, supershiny, sparkling wafers all in a row in polishing-machine carriers, awaiting inspection.*

pencil-shaped silicon crystal, so at the ingot's first stop outside of the furnace room it is cropped—both its shoulders and its tail are amputated, and it becomes a conforming, standardized cylinder of glass. To find out whether the resistivity of the crystal meets the specifications of the customer, another operator applies a four-point probe (looks like a four-needle stereo cartridge) to both cropped ends of the cylinder. If both ends are within the acceptable range, then surely the middle of the cylinder must be too.

After being cropped and probed, the cylinder, whose sides are a bit ripply from the natural freezing processes, is placed on a grinder between two pressure plates, each pulling towards the middle with about four hundred pounds of pressure. As the grinder spins the crystal along its vertical axis, a round doughnut-shaped diamond cup wheel moves across the ingot, standing off about an eighth of an inch all the way around.

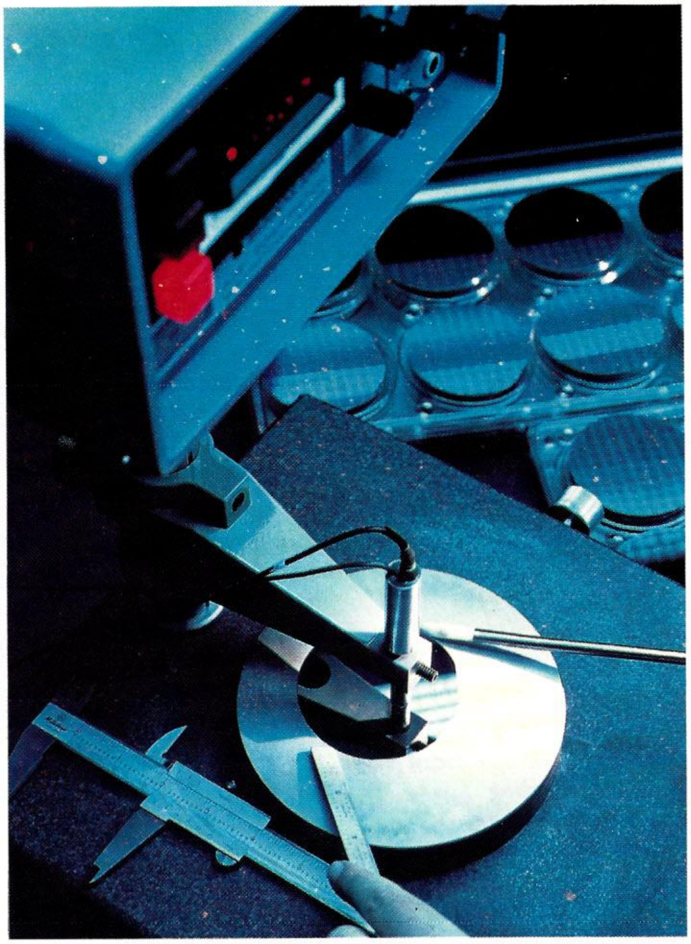
Once ground into a gray matte, smooth cylinder, the ingot travels to the x-ray station, where yet another operator takes a Laue radiograph to find where the internal rays are. When the operator has found the crystal's best side, it is marked and a "flat" is ground down along the whole length of the crystal parallel to a spine. Thus, the characteristic wafer shape, a circle with one side sawn off. Sometimes a secondary flat is ground at forty-five degrees to the first flat.

The flats identify the internal structure of the crystal and each of its wafers. When the crystal is sliced up into wafers, each wafer will break, if stressed, parallel to its internal spine. After the customer has drawn two-hundred-fifty-odd chips on the wafer, he will scratch a grid between the edges of each chip, and then break the wafer into two hundred fifty little chips, the way a glass cutter scratches and then breaks a pane of glass. If the chips have been printed on the wafer so that they run parallel and perpendicular to the internal latticework, they will break very predictably. If not, they won't. The flat simply tells the customer where to print the chips on the wafer.

*Something there is that abhors a pencil-shaped silicon crystal, so its shoulder and tail are amputated.*

All of this grinding has left microscopic scratches on the surface of the cylinder, damage that could propagate into the heart of the ingot, the way a sharp blow on a window causes a small web of cracks which travel through the whole pane of glass. A bath in a blend of hydrofluoric and nitric acid etches off all irregularities and returns the native high gloss to the silicon's surface.

The cylinder now proceeds to the wafer cutting area. Its flat becomes an expedient surface for applying epoxy mounting compound. The pink adhesive is slathered along the flat, and then the cylinder is set vertically onto the chuck of the saw as if the crystal had been indeed plastered to the bedpost overnight with a giant wad of bubble gum. The ingot is lowered into the hole of a doughnut-shaped thin, flexible saw, about fourteen inches across and about four millimeters thick, just a little larger than a record and almost as flexible as a piece of heavy-duty cooking aluminum foil. Along the edge of the hole, a five-inch band of cutting diamonds are massed. The ingot on its chuck moves back and forth between the hole and the middle of the saw, the diamonds abrading off two-inch wafers into ten to twelve mil thicknesses, 1/91 of an inch; three-inch wafers into fourteen to sixteen mil thicknesses, 1/67 of an inch; and four-inchers into nineteen to twenty-one mils, about 1/50 of an inch. A mil equals twenty-five microns, or 1/1000 of an inch. As the ingot is sliced, like so much bread in a bakery, the wafers don't fall off. They stay embedded in the epoxy. Every fifteen wafers, or so, the machine automatically stops, and the operator lifts off the waiting wafers. Occasionally, the



*Virgin wafers must pass rigid inspection before they are ready for masking, imprinting, and developing.*

operator dresses the blade by running a three hundred grit aluminum oxide abrasive stick along the diamond edge of the saw, melting the nickel which holds the diamonds so that the diamonds can realign themselves to a fresh cutting edge.

The edges of the newly cut wafers are sharp and have random tiny ridges which could interfere with the alignment of masks during the chip printing processes, collect a buildup of photoresist chemicals, and lead to crazing. So, using a jealously guarded proprietary process, Siltec sands or abrades the edges of each wafer so that it is as flat as computer technology can manage it—this year.

All this sawing and sanding kicks up silicon dust. The dust is trapped in the water coolant mixture splashed on the saw blade to keep it cool, resulting in wafers covered with a gray mud. Wafers are loaded, twenty-five at a time, into baby-blue plastic cassettes and then dipped for about thirty seconds into the reddish brown hydrofluoric-nitric acid bath. As the wafers enter the bath, the acid wells up threateningly, the surface of the liquid billows with white foam, and brown fumes escape from the surface and up to a strategically placed vent, to be cleansed in a closed system before they are released into the ambient air. The acid is rinsed off automatically in an adjacent tub of water. All etching pools are completely enclosed in the same incubator-like system.

Once again, resistivity must be verified, this time on a computerized machine designed by Siltec to sort twenty-five wafers a minute for resistivity and thickness. Now travelling in batches with hundreds of similar ilk, the wafers go to the polishing area where they receive a mirror finish on one side. The key to polishing lies in mounting the wafers flat so that as they are buffed, they are not bent, warped, or bowed. Any concavity or bump in a wafer, no matter how microscopic, decreases the sharpness of focus when the chip circuitry is imprinted in photoresist technique on the wafer. Most manufacturers have been mounting the wafers with waxes which leaves air bubbles and bulges. As the wafers are pressed during the buffing process they are pushed to conform to the shape of the air bubbles and bulges, distorting the wafer.

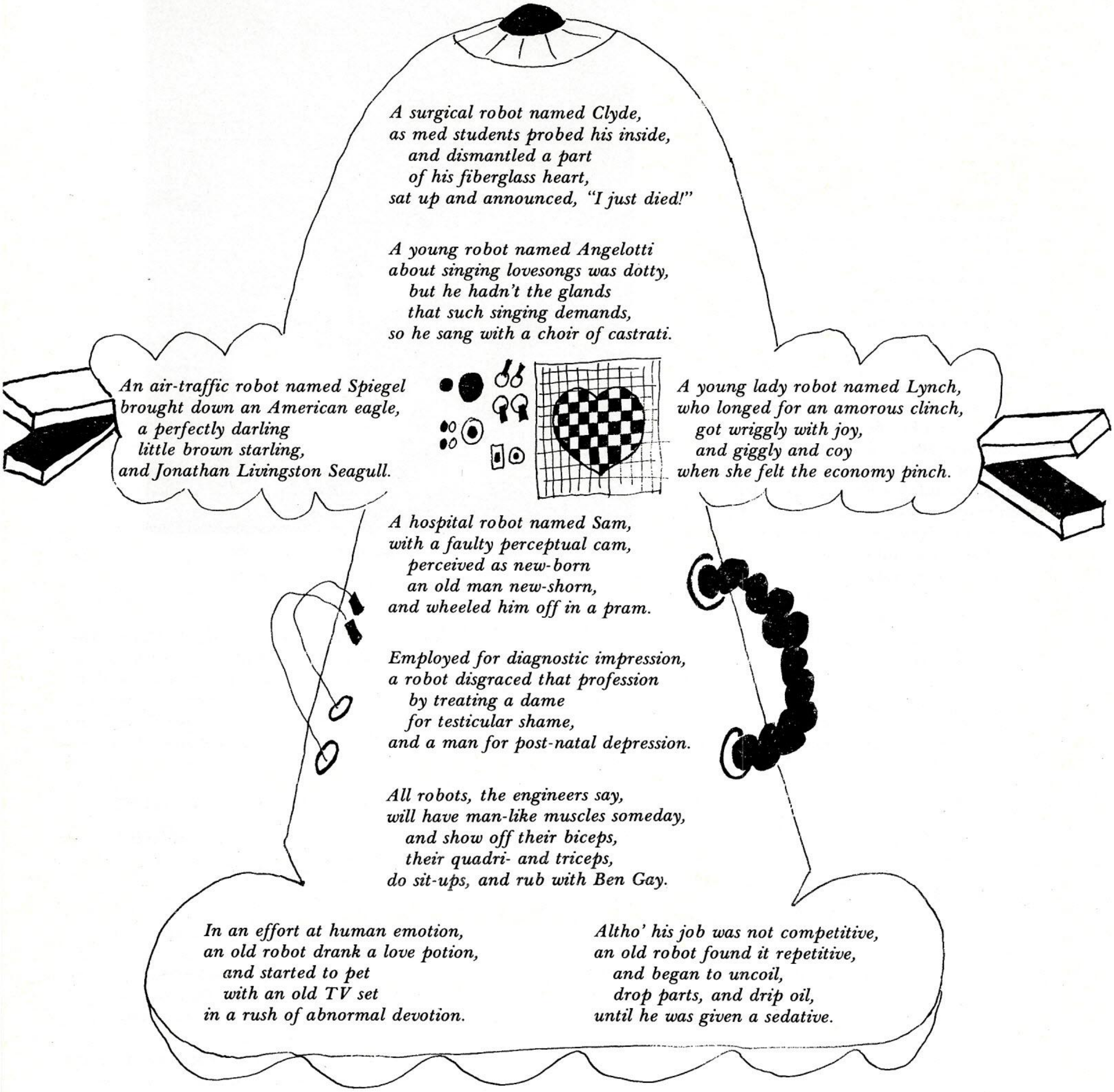
Siltec's adhesive is a dark secret, but they claim the wafers come out "super-flat." The polishing machine looks like a giant photocopying machine. The lid holds four disks, each with six wafers. When the lid is closed, the disks revolve in one direction, and the three-foot diameter polishing pad lying below spins in the other direction.

*As the wafers enter their bath, the acid wells up threateningly, the liquid billows with foam, the brown fumes rise.*

Once polished to a brilliant sheen, the wafers are washed, "demounted," inspected for the perfection of their mirror finish, their flatness, their resistivity, again, and their thickness. Each virgin wafer of scientifically propagated silicon crystal stands ready to be masked, imprinted, and developed. Once scribed and broken into two or three hundred individual chips, they will be ready to slip silently into the functioning world of your computer. ▼

# *Robotic Verse*

## by Gloria Maxson



*A surgical robot named Clyde,*
*as med students probed his inside,*
*and dismantled a part*
*of his fiberglass heart,*
*sat up and announced, "I just died!"*

*A young robot named Angelotti*
*about singing lovesongs was dotty,*
*but he hadn't the glands*
*that such singing demands,*
*so he sang with a choir of castrati.*

*An air-traffic robot named Spiegel*
*brought down an American eagle,*
*a perfectly darling*
*little brown starling,*
*and Jonathan Livingston Seagull.*

*A young lady robot named Lynch,*
*who longed for an amorous clinch,*
*got wriggly with joy,*
*and giggly and coy*
*when she felt the economy pinch.*

*A hospital robot named Sam,*
*with a faulty perceptual cam,*
*perceived as new-born*
*an old man new-shorn,*
*and wheeled him off in a pram.*

*Employed for diagnostic impression,*
*a robot disgraced that profession*
*by treating a dame*
*for testicular shame,*
*and a man for post-natal depression.*

*All robots, the engineers say,*
*will have man-like muscles someday,*
*and show off their biceps,*
*their quadri- and triceps,*
*do sit-ups, and rub with Ben Gay.*

*In an effort at human emotion,*
*an old robot drank a love potion,*
*and started to pet*
*with an old TV set*
*in a rush of abnormal devotion.*

*Altho' his job was not competitive,*
*an old robot found it repetitive,*
*and began to uncoil,*
*drop parts, and drip oil,*
*until he was given a sedative.*

From *Glorobots* by Gloria Maxson. Copyright © 1977 by Gloria Maxson. The book is available from the author, 13602 Cullen Street, Whittier, CA 90605.

## RADIO SHACK RUNS WITH Z-80

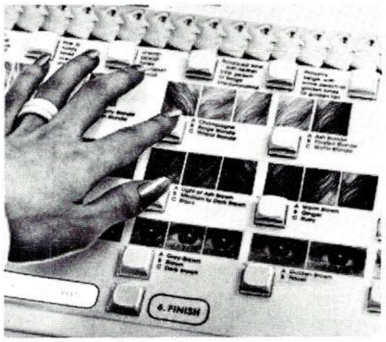
Radio Shack is for real with its realistically priced (if not so named) micro. The ready-to-plug-in-and-run TRS-80 sells for $599.95 complete with a fifty-three-key keyboard, regulated power supply, interfaced cassette recorder, and twelve-inch video display monitor. As if the low price isn't enough, the real marketing con is the instant availability of five prerecorded programs. For a complete library Radio Shack is still the premier purveyor of ready-to-run systems with something to run.

Applications software so far includes the demonstration blackjack and backgammon cassette that comes with the unit as well as a payroll program, a math education program, and a personal finance program. More on the way. All on prerecorded cassettes. At your local Radio Shack, or write:

*Radio Shack*
*2617 West Seventh St.*
*Fort Worth, TX 76107*

## COSMETOLOGICAL MICRO

Fingernail polish for your micro? Not really. But to show how all pervasive the micro is about to become—and how someone with the right applications idea can carve out a comfortable little commercial niche for himself in a market that didn't even exist yesterday, consider the Computerized Cosmetic Selector System developed by Emik Avakian.



Designed for the Helena Rubenstein Company, the "Skin Life Instant Beauty Analyzer" takes numeric reference values in the form of the consumer's personalized input of hair and eye color, skin tone, and skin type, stirs them up in the CPU with a complete line of the company's cosmetics, and outputs a selection of suitable cosmetological choices.

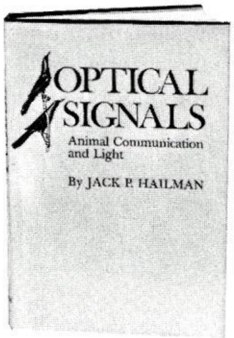
Sales gimmick? Certainly. Will it sell the cosmetics? Almost as certainly. So who out there is going to make his fortune on a point-of-purchase micro for the milliners' trade?

*Avakian Systems Corp.*
*70 Oakwood Drive*
*Glastonbury, CT 06033*

## EYES FOR THE COMPUTER

More than once technology has found itself benefiting from mimicking nature. Research in robotics or in artificial intelligence that fails to take into account the workings of already operative natural systems would seem almost by definition to be doomed to failure. Even on the relatively simple level of communications technology, biological emulation offers a most fertile area for exploration. All of this brings us to *Optical Signals: Animal Communication and Light,* a volume that might not normally come to the attention of the computerist.

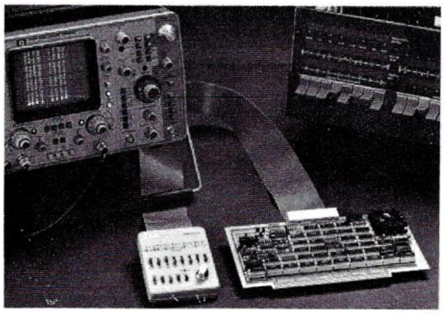
Delving into semiotics, information theory, ethology, and social communications, the author, Jack P. Hailman, investigates the physics of light from the viewpoint of information transmission, showing how it permits the relaying of information while at the same time it



imposes very specific limitations. In a McLuhanesque twist to signal theory, Hailman also considers the actual message transmitted and its influences on the form and parameters of the signals themselves. The presentation alone of the conceptual framework behind the design characteristics of optical signals is worth the price of the book for anyone involved with communications technology.

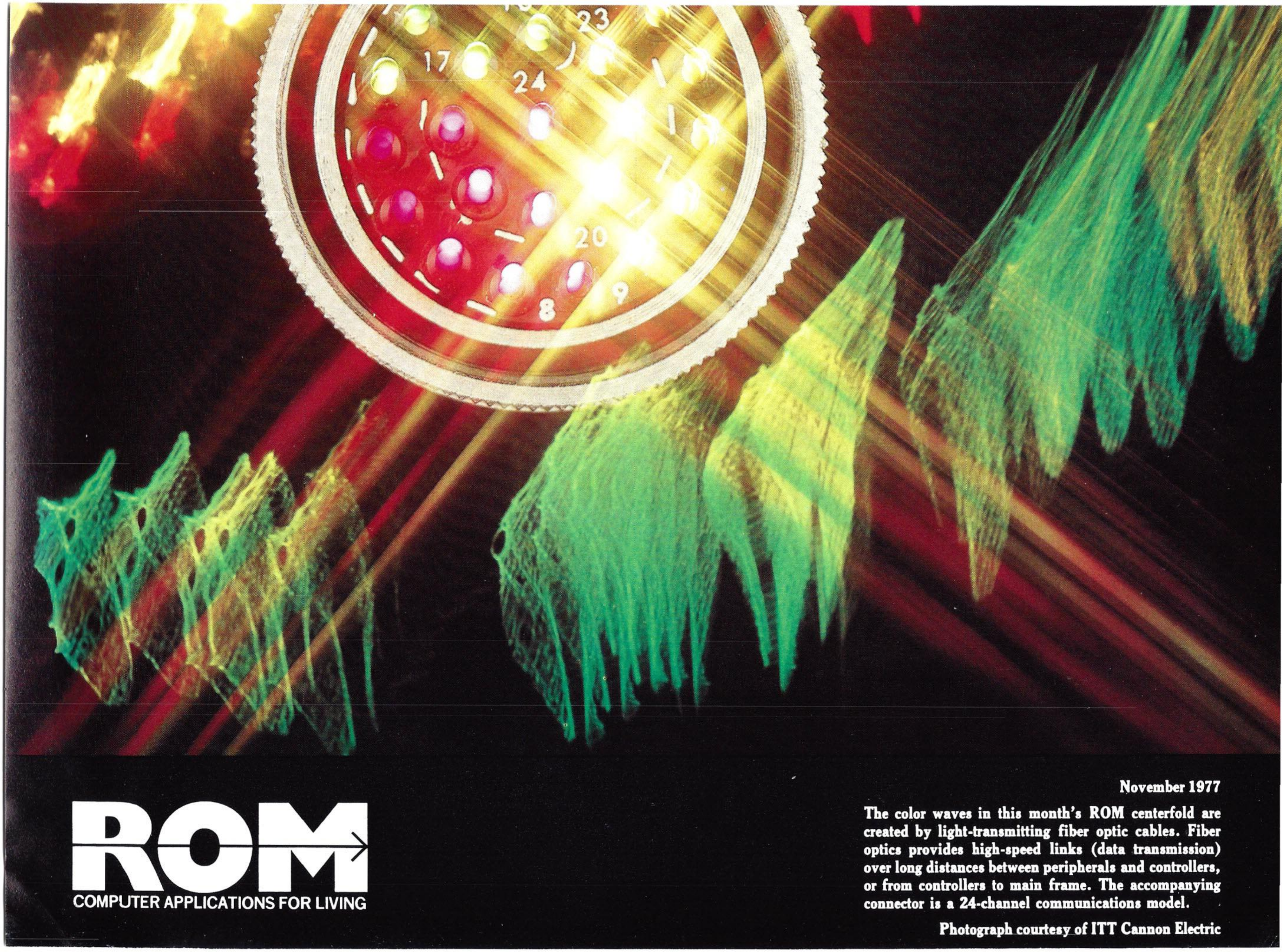
*Indiana University Press*
*Bloomington, IN 47401*
*$15.*

## GRABBING THE S-100



*Paratronics, Inc.*
*800 Charcot Ave.*
*San Jose, CA 95131*
*Kit $369. Assembled $449.*

If you're looking for some way to cash in on your interest in computing, servicing the mushrooming myriad of machines may well be the answer. Of course you'll need some tools—for example, the 150 "Bus Grabbing" Logic Analyzer, which interfaces electrically and mechanically with the S-100 bus. Among other things this dedicated model will monitor all fifty-six key signals through one plug connection to the board. No more tripping over your probes. Many other goodies.

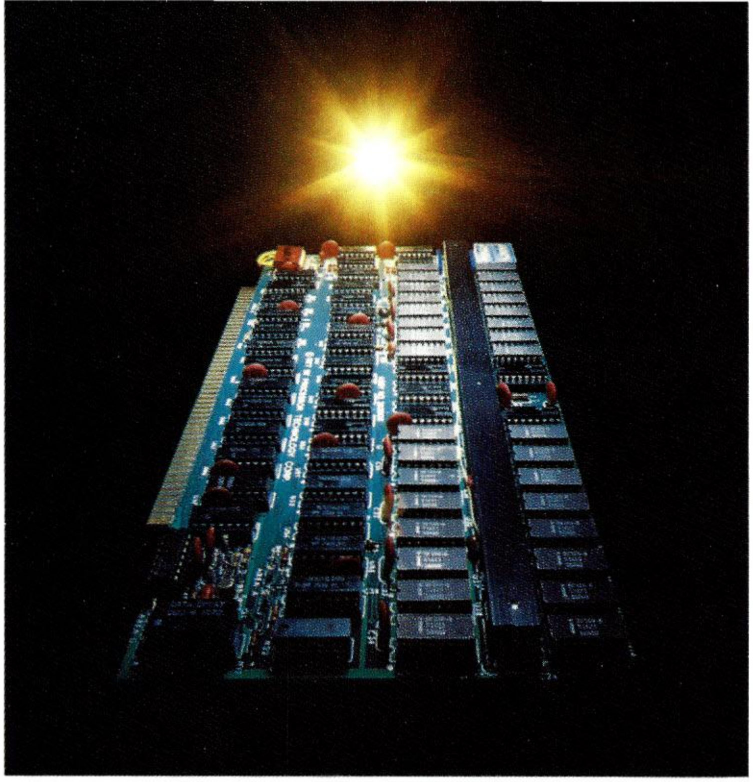


**ROM**
COMPUTER APPLICATIONS FOR LIVING

**November 1977**

The color waves in this month's ROM centerfold are created by light-transmitting fiber optic cables. Fiber optics provides high-speed links (data transmission) over long distances between peripherals and controllers, or from controllers to main frame. The accompanying connector is a 24-channel communications model.

Photograph courtesy of ITT Cannon Electric



# The end of Kit-Kits.

## The end of bad solder joints, heat damaged components and sick IC's. Introducing the Semikit. Item 1, a 16KRA Memory Board, $369.

Let's face it. Loading and soldering PC Boards is not much fun for the kit builder. Even more important, it's the place where most of the trouble gets introduced. The real fun and education comes in running and testing boards.

### Now the Semikit with fully tested IC's.

At the price of a kit, Processor Technology Corporation introduces the Semikit. It's a fully stuffed, assembled and wave soldered PC Board loaded with IC's that have gone through Q.C. and final check-out (a first in the industry).

We leave you the fun of testing with our fully documented set of instructions. We do the production tasks of loading, wave soldering and inspecting the boards. You do the more interesting and time consuming chore of testing and burning-in the boards.

The result is one sweet deal for both of us. You get a board where the primary causes of damage (poor solder joints, excess solder and bad IC's) are virtually eliminated. You get a board of highest professional quality. And we get the business!

### The 16KRA Memory Board's at your dealer now.

Your Processor Technology dealer has the first Semikit, a 16KRA Memory Board, in stock and ready to go right now. You can take it home tonight for $369 as a Semikit or for $399 fully assembled, tested and burned-in.

You'll have a 16,384 byte memory with a better price performance ratio than anything on the market today. Now you can afford to add quality, high density memory to your system for remarkably little. And you can add enough to solve complex computing problems right in the main frame.

The memory features invisible refresh. There's no waiting while the CPU is running. Worst case access time is 400 nsec. Each 4,096 word block is independently addressable for maximum system flexibility. Power is typically 5 watts, the same as most single 4K memory modules. Back-up power connection is built-in.

### Other Semi's are coming your way.

The 16KRA Memory is Processor's first step in adding more fun, capability and reliability to your computer system at lower cost. Other modules are on the way to your dealer now. Come on down today.

Or you may contact us directly. Please address Processor Technology Corporation, Box O, 7100 Johnson Industrial Drive, Pleasanton, California 94566. Phone (415) 829-2600.

**Processor Technology**

# See Sol here...

### ALABAMA

ICP, Computerland
1550 Montgomery Hwy.
Birmingham, AL 35226
(205) 979-0707

### ARIZONA

Byte Shop Tempe
813 N. Scottsdale Rd.
Tempe, AZ 85281
(602) 894-1129

Byte Shop Phoenix
12654 N. 28th Dr.
Phoenix, AZ 85029
(602) 942-7300

Byte Shop Tucson
2612 E. Broadway
Tucson, AZ 85716
(602) 327-4579

### CALIFORNIA

The Byte Shop
1514 University Ave.
Berkeley, CA 94703
(415) 845-6366

Computer Center
1913 Harbor Blvd.
Costa Mesa, CA 92627
(714) 646-0221

DCI Computer Systems
4670 N. El Capitan
Fresno, CA 93711
(209) 266-9566

Bits 'N Bytes
679 S. State College Blvd.
Fullerton, CA 92631
(714) 879-8386

The Byte Shop
16508 Hawthorne Blvd.
Lawndale, CA 90260
(213) 371-2421

Opamp/Computer
1033 N. Sycamore Ave.
Los Angeles, CA 90038
(213) 934-3566

The Computer Mart
624 West Katella #10
Orange, CA 92667
(714) 633-1222

Byte Shop
496 South Lake Ave.
Pasadena, CA 91101
(213) 684-3311

Micro-Computer
Application Systems
2322 Capitol Avenue
Sacramento, CA 95816
(916) 443-4944

The Computer Store
of San Francisco
1093 Mission Street
San Francisco, CA 94103
(415) 431-0640

Byte Shop
321 Pacific Ave.
San Francisco, CA 94111
(415) 421-8686

The Byte Shop
2626 Union Avenue
San Jose, CA 95124
(408) 377-4685

The Computer Room
124H Blossom Hill Rd.
San Jose, CA 95123
(408) 226-8383

The Byte Shop
509 Francisco Blvd.
San Rafael, CA 94901
(415) 457-9311

The Byte Shop
3400 El Camino Real
Santa Clara, CA 95051
(408) 249-4221

Recreational Computer
Centers
1324 South Mary Ave.
Sunnyvale, CA 94087
(408) 735-7480

Computer Components
5848 Sepulveda Blvd.
Van Nuys, CA 91411
(213) 786-7411

The Byte Shop
2989 North Main St.
Walnut Creek, CA 94596
(415) 933-6252

Byte Shop
14300 Beach Blvd.
Westminster, CA 92683
(714) 894-9131

### COLORADO

Byte Shop
2040 30th St.
Boulder, CO 80301
(303) 449-6233

Byte Shop
3464 S. Acoma St.
Englewood, CO 80110
(303) 761-6232

### FLORIDA

Byte Shop of Miami
7825 Bird Road
Miami, FL 33155
(305) 264-2983

Microcomputer
Systems Inc.
144 So. Dale Mabry Hwy.
Tampa, FL 33609
(813) 879-4301

### GEORGIA

Atlanta Computer Mart
5091-B Buford Hwy.
Atlanta, GA 30340
(404) 455-0647

### ILLINOIS

Champaign Computer
Company
318 N. Neil Street
Champaign, IL 61820
(217) 359-5883

itty bitty machine co.
1316 Chicago Ave.
Evanston, IL 60201
(312) 328-6800

itty bitty machine co.
42 West Roosevelt
Lombard, IL 60148
(312) 620-5808

### INDIANA

The Data Domain
406 So. College Ave.
Bloomington, IN 47401
(812) 334-3607

The Byte Shop
5947 East 82nd St.
Indianapolis, IN 46250
(317) 842-2983

Computers Unlimited
7724 East 89th Street
Indianapolis, IN 46256
(317) 849-6505

The Data Domain
7027 N. Michigan Rd.
Indianapolis, IN 46268
(317) 251-3139

### IOWA

The Computer Store
of Davenport
616 West 35th Street
Davenport, IA 52806
(319) 386-3334

### KENTUCKY

The Data Domain
3028 Hunsinger Lane
Louisville, KY 40220
(502) 456-5242

### MICHIGAN

The Computer Store
of Ann Arbor
310 East Washington
Ann Arbor, MI 48104
(313) 995-7616

Computer Mart
of Royal Oak
1800 W. 14 Mile Rd.
Royal Oak, MI 48073
(313) 576-0900

General Computer Store
2011 Livernois
Troy, MI 48084
(313) 362-0022

### MINNESOTA

Computer Depot, Inc.
3515 W. 70th St.
Minneapolis, MN 55435
(612) 927-5601

### NEW JERSEY

Hoboken Computer Works
No. 20 Hudson Place
Hoboken, NJ 07030
(201) 420-1644

The Computer Mart
of New Jersey
501 Route 27
Iselin, NJ 08830
(201) 283-0600

### NEW YORK

The Computer Mart
of Long Island
2072 Front Street
East Meadow, L.I. NY 11554
(516) 794-0510

The Computer Shoppe
444 Middle Country Rd.
Middle Island, NY 11953
(516) 732-4446

The Computer Mart
of New York
118 Madison Ave.
New York, NY 10001
(212) 686-7923

The Computer Corner
200 Hamilton Ave.
White Plains, NY 10601
(914) 949-3282

### OHIO

Computer Mart of Dayton
2665 S. Dixie Ave.
Dayton, OH 45409
(513) 296-1248

### OREGON

Byte Shop Computer Store
3482 SW Cedar Hills Blvd.
Beaverton, OR 97005
(503) 644-2686

The Real Oregon
Computer Co.
205 West 10th Ave.
Eugene, OR 97401
(503) 484-1040

Byte Shop Computer Store
2033 SW 4th Ave.
Portland, OR 97201
(503) 223-3496

### PENNSYLVANIA

Byte Shop of
Delaware Valley
1045 Lancaster Pike
Bryn Mawr, PA 19010
(215) 525-7712

### RHODE ISLAND

Computer Power, Inc.
M24 Airport Mall
1800 Post Rd.
Warwick, RI 02886
(401) 738-4477

### TEXAS

Computer World
926 N. Collins
Arlington, TX 76011
(817) 469-1502

Byte Shop
3211 Fondren
Houston, TX 77063
(713) 977-0664

Computertex
2300 Richmond Ave.
Houston, TX 77006
(713) 526-3456

Interactive Computers
7646½ Dashwood Rd.
Houston, TX 77036
(713) 772-5257

Neighborhood Computer
Store
#20 Terrace Shopping Center
4902 - 34th Street
Lubbock, TX 79410
(806) 743-2787

The Micro Store
634 So. Central
Expressway
Richardson, TX 75080
(214) 231-1096

### VIRGINIA

The Computer Systems
Store
1984 Chain Bridge Rd.
McLean, VA 22101
(703) 821-8333

Media Reactions Inc.
11303 South Shore Dr.
Reston, VA 22090
(703) 471-9330

The Home Computer Center
2927 Virginia Beach Blvd.
Virginia Beach, VA 23452
(804) 340-1977

### WASHINGTON

Byte Shop Computer Store
14701 N.E. 20th Ave.
Bellevue, WA 98007
(206) 746-0651

The Retail Computer Store
410 N.E. 72nd
Seattle, WA 98115
(206) 524-4101

### WISCONSIN

Madison Computer Store
1910 Monroe St.
Madison, WI 53711
(608) 255-5552

The Milwaukee
Computer Store
6916 W. North Ave.
Milwaukee, WI 53213
(414) 259-9140

### CANADA

Trintronics
160 Elgin St.
Place Bell Canada
Ottawa, Ontario K2P 2C4
(613) 236-7767

First Canadian
Computer Store Ltd.
44 Eglinton Ave. West
Toronto, Ontario M4R 1A1
(416) 482-8080

The Computer Place
186 Queen St. West
Toronto, Ontario M5V 1Z1
(416) 598-0262

Pacific Computer Store
4509-11 Rupert St.
Vancouver, B.C. V5R 2J4
(604) 438-3282

**ProcessorTechnology**

# The Kit
# Part Three:

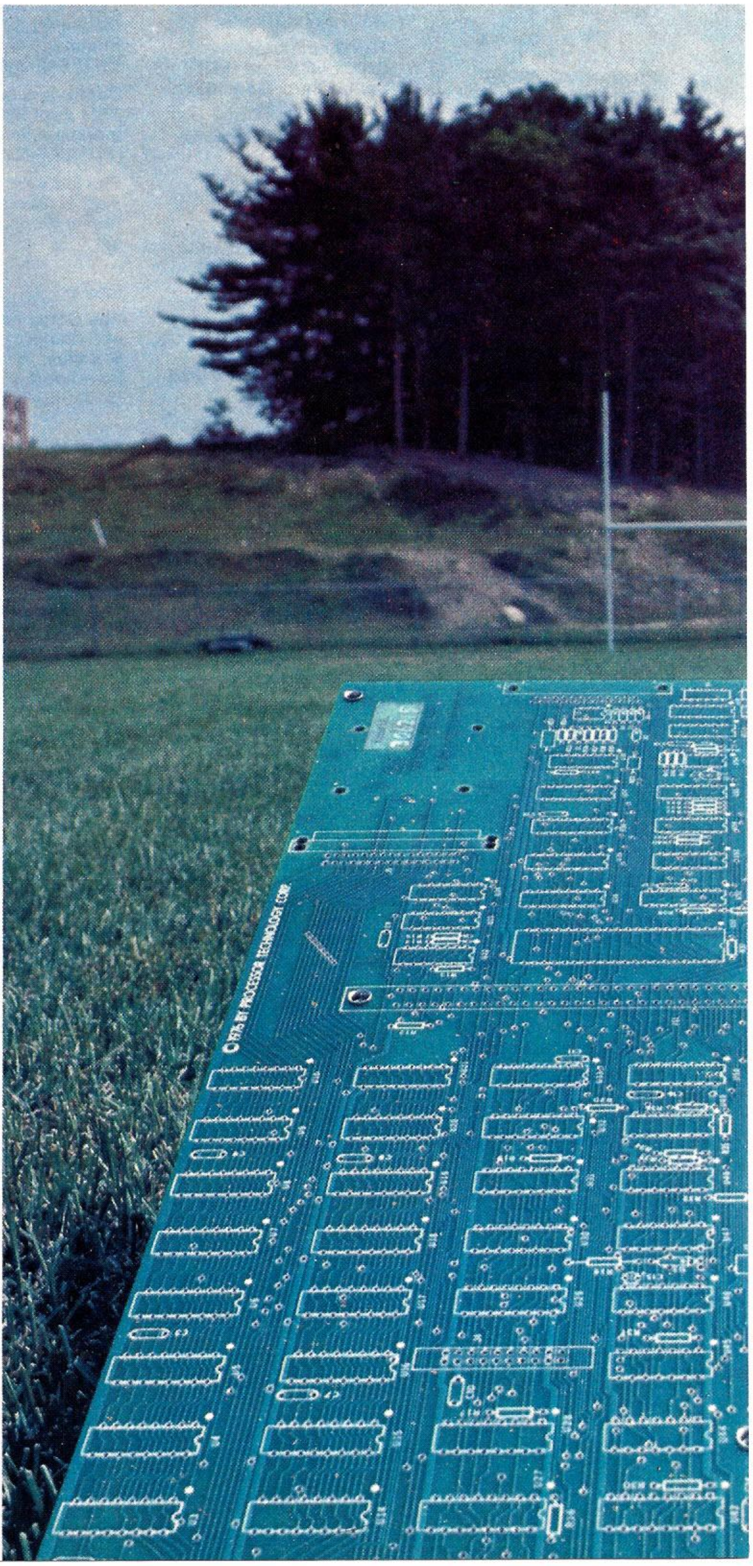
by Richard

The main board for Sol spread out in front of me like a football field. It had been hidden in a separate carton which I'd overlooked, thinking it was part of the cabinet enclosure. Now, faced with its broad expanse filled with an incalculable number of future soldering points, I was glad I hadn't seen it when starting the project, or I'd probably have been intimidated enough to leave the whole affair unassembled in the closet. As it was now, I was hooked. Onward and upward.

After checking the board visually, which was akin to looking at a map of Tokyo to ascertain if the cartographer had left out any alleys or not, though you'd never been west of Hawaii, I went on to search out the Sol-PC assembly drawing. In the instructions I was told that this key document would be located in Section IX. This, however, was labeled "Software," and so I searched elsewhere. The drawing was duly located in Section X, clearly marked X, etc.

Actually, by now I'd begun to know my way around the boards to some extent. The fact that I started the project by assembling the 8KRA memory board was purely an accident a la the U. S. Postal System, but in its own way it was a helpful introduction. Rather than spending an hour or so looking for the five-volt and twelve-volt buses on the huge main board as I would have with no previous experience, I searched them out with relative ease, even giving a sage nod when I noticed them clearly marked J10 while the instruction book gave no indication of the crucial cartographic coordinate.

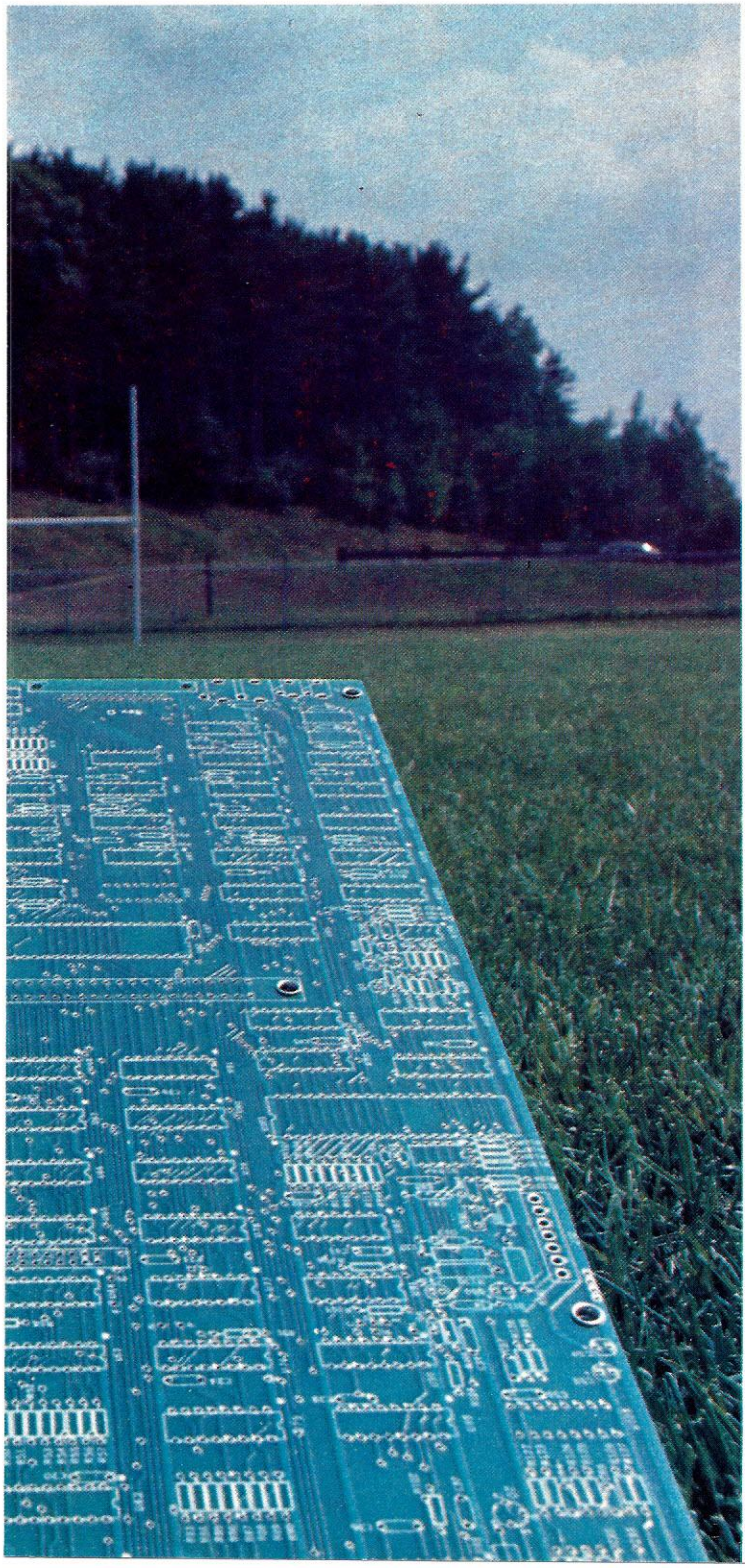
My self-satisfied reverie came to an abrupt end. The next step told me that the personality module would be required for testing the Sol-PC during assembly, and so I was to put the big board aside and proceed to Section IV



Photographed by Thomas Hall

# and I
# Personality Plus

### W. Langer



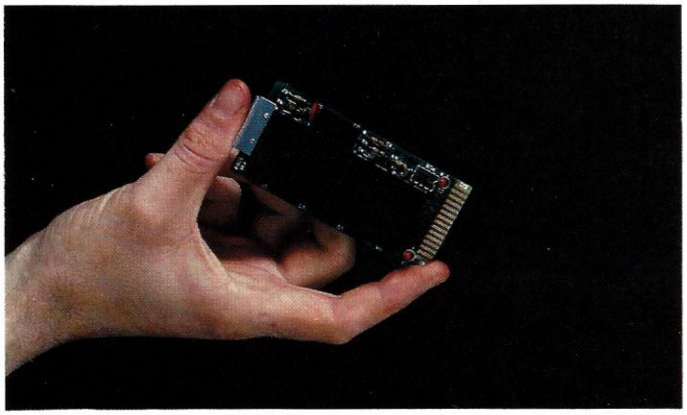
in search of my computer's personality. Which I did.

At first I thought they'd failed to pack the personality module with my kit. Couldn't find the board. This, it turned out, was due to a distortion of my size perspective after facing the main PC board. With one that size, the computer had to have a lot of personality. Surprise, character couldn't be its main strength, the personality board was about half the size of a shaved ace of spades.

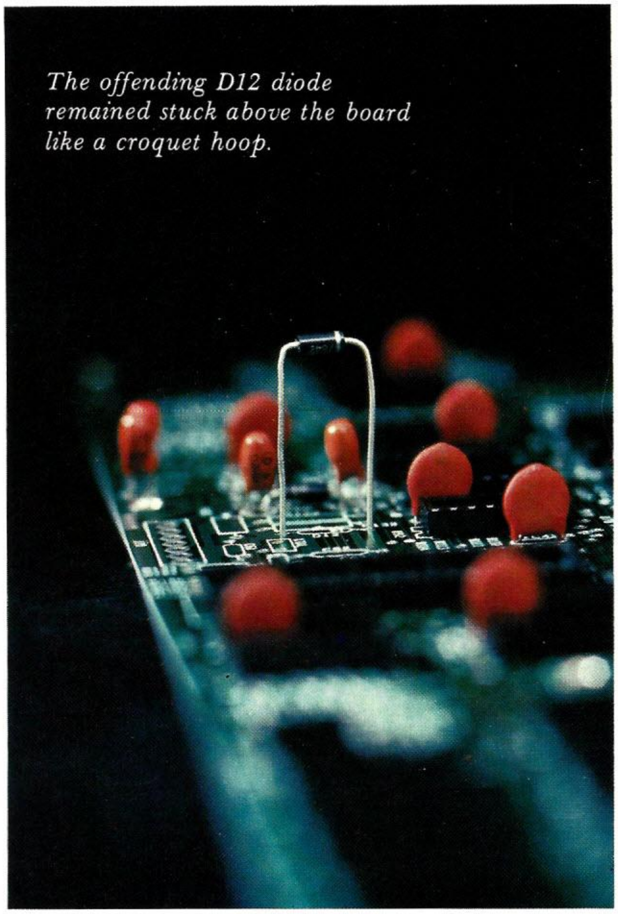
Only four sockets to solder. Small time stuff. The resistors were something else again. Four of them had to be mounted vertically on the board—standing on their heads, so to speak. Did this alter their resistance, or was it simply a space-saving trick? Although the latter didn't seem likely since there was room for the other resistors in their normal prone fashion, it seemed to be the only reasonable explanation. Explanations, explanations, why don't they give explanations? Then again they didn't promise me a basic course in electronics. Come to think of it, I'm not sure they promised me anything.

Shortly after the headstanding act, I came across another novelty. Something called "jumpers." As I was trying to find out why the resistors were by necessity of a headstanding inclination, I noticed a note on the assembly drawing, "Jumpers labeled 'J1' must be installed on PM5204 modules using National 5204Q PROMs." Did I even have to look? I mean, how could it be any other way? Still, I couldn't resist. Sure enough, I had the 5204Q PROMs. By now I was getting wise—someone had left out some connections on the board and now they expected me to cover up. Well, all right, I was willing. But what kind of wires does one use for jumpers? I mean, if the resistance was crucial enough to possibly stand things on

Character couldn't be its main strength, the personality board was about half the size of a shaved ace of spades.



The offending D12 diode remained stuck above the board like a croquet hoop.



their head, how much extra resistance would I introduce by using the wrong diameter wire?

One problem at a time. The next step was to check for shorts. The moment of truth. I whipped out my trusty, if not yet fully conquered ohmmeter. Nothing. Somewhere in that little board I'd managed to build in a short, or at least so it appeared. Taking the personality down to my local friendly computer store, however, I discovered I'd had the ohmmeter on the wrong scale setting. Sol's personality was fine; it was just mine that lacked something.

Returning home, I finished up the diminutive module. For jumpers I decided some leftover leads previously trimmed off other components would do. The instructions said, "#24 bare wire may be used." Since the leads held all the other components down, they'd work as jumpers as well, I reasoned. This left only the socket location marked PGM, to which the instructions made no reference. Figuring my Sol was one of the more mature models, and the Parental Guidance Module, as I thought of it, was not necessary, I proceeded to tackle the football field of a main board. Sure were a lot of pins to solder.

Socket after socket slipped into place. Here was visible progress on a massive scale. Elated, I zipped along with my trusty soldering iron, till suddenly I was confronted with an empty hole in a long line of pins I'd soldered. Scanning the board edgewise, I discovered the problem. One of the pins had bent over when I'd positioned the socket.

Now faced with half the pins already soldered in place, I discovered the pleasure of *unsoldering*. What a pain. First I tried to simply melt the solder and pull the plug. Ha! With six pins I had to be quicker with the iron than a short order cook flipping flapjacks after an unexpected busload pulled in for breakfast. And I wasn't.

Then I heard about soldering wicks. Turns out to be like the wick for an alcohol lamp but it's woven out of copper. Having once lit an alcohol lamp for warmth using frozen fingers after a day hacking my way through charging armies of mosquitoes, I may add it was a lot easier than getting all the solder out of a hole in the board using this contraption.

Later someone suggested a desoldering tool. As it is, I never did get the offending socket removed. But things loosened enough so I could get the bent lead straightened out and in place.

Capacitor time again. A long line of them. C1, C2, C3, C4, C5, C6, C7, C8.... More on the next page. All simple enough until it was time for another check with the ohmmeter. Something was shorting out somewhere. I turned the board over and began the painful task of threading my way through the minefield of solder joints. There must have been a thousand of them at least. I went over each one. Nothing. I resorted to a Sherlock Holmes-like magnifying glass. Well, it wasn't elementary, my dear Watson; still nothing. The only hint the manual gave me, in case I was unfortunate enough to suffer from continuity at this stage, was "Find and correct the problem before proceeding to Step 4." Wonderful. How about at least giving me a hint as to what the problem might be? All I could think of was a short somewhere. But after my third pass over the board, I was convinced that was not where the problem lay. All right, then, perhaps I put one of the capacitors in the wrong place. Unlikely, since they were clearly labelled C1, C2, C3. . . . I turned the page. C9 did not follow C8, C10 followed C8. Back to my newly acquired skill of unsoldering.

Even with the offending C9 removed the board suffered from the inappropriate continuity. I went out for a quiet pipeful by the dam. Then in one of those now-you-see-it-now-you-don't flipflops, I realized I'd failed once again to understand what I was measuring when using the ohmmeter. Because the needle moves from left to right, habit led me to think that as it moved, the value increased. All the instruments I'm familiar with—the gas gauge on my car, the speedometer, etc.—have the zero on the left and the high scale on the right. As the needle moves up, the values move up. On the ohm scale the zero is on the right and high scale on the left. *As the needle moves up, the reading goes down.* Just watching for the needle movement, I'd been blind to the numbers right in front of my eyes. Well, the fish couldn't see me blush; that was the best I could say for the matter.

I went back to the uneventful task of putting in more capacitors. Although I hadn't quite figured out their coding, the process of elimination served me well. The real chore was pushing them back and forth through the holes of the board in order to remove the wax coating on the leads as the instruction manual repeatedly exhorted me to do.

Then I thought I remembered, somewhere from my distant past, something about alcohol's being a wax solvent. I left my study momentarily, headed for the house, and picked up a Scotch on the rocks. Putting a few drops on my fingers, I wiped the leads without much success. Still, the Scotch didn't go to waste. Particularly when it came time to delve into the diode bag.

Diode D12 simply wouldn't fit. The leads refused to slide into the holes; the diode remained stuck above the board like a croquet hoop. It looked as if I had switched games.

Below the instructions on installing the diodes, I came across the note: "The leads of D12 and its mounting holes are a snug fit. Take care when installing this diode."

Snug fit indeed. It didn't fit, so how did one take care installing it? (I wonder why they always put these notes after the problem rather than warning the hapless hobbyist in advance. It could save a lot of frustration.)

Sanding the leads seemed hopeless. Enlarging the holes did not seem wise, since by holding the board up to the light one could see that the metal linings went from one side to the other and in some cases they appeared to act as bridges between the top side and the bottom. I called a friend; he told me the best solution would be to solder thinner leads to the diodes. Elegant in thought, perhaps, but not in execution. D12 now has warts on its leads.

A run of resistors and a covey of capacitors followed uneventfully before it was time to install the 14.318 MHz crystal. With visions of diamonds dancing through my head, I searched in vain for said sparkler. As a last resort I followed two leads into a sponge wrapper and came up with a little tin box. I guess one doesn't get to look at its sparkling beauty.

After a couple of ICs, some jumpers, and my old friend the Molex were installed, I was told to check the clock circuits. At that stage I didn't even realize my would-be computer could tell time.

Checking the clock circuit offered two choices. Either using an oscilloscope or building a probe to use with the ohmmeter. The oscilloscope check was listed first, so it seemed to be preferred. Besides, I'd given myself enough problems with the ohmmeter. On the other hand, an oscilloscope wasn't exactly something I kept around the tractor barn.

So I called up my local Taylor Rental. Why not? I'd once rented a ditch digger with a broken left wheel sprocket there. As a result, my power line from the house to the study runs in a series of half ellipses rather than a straight line. At the time, I'd seen some arc welders and other electrical equipment there, between the rose-patterned china for large unexpected parties, so why not an oscilloscope? Well, I don't know why not. They didn't know why not. But they didn't.

Looking out the window, I could see the gray sex-linked rooster strutting among the brown hens as the sun set over the pond. No doubt about it. The barn did need painting. But as the twilight dimmed, less and less of the peeling coat of red buttermilk was noticeable.

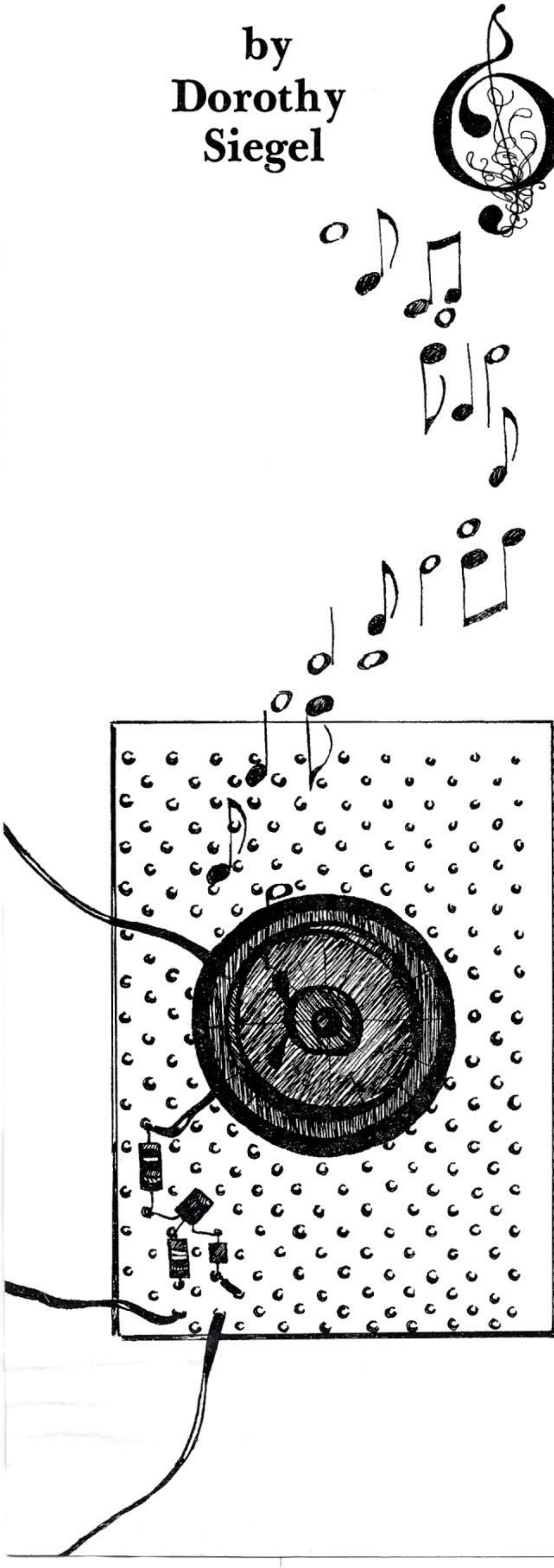
The dogwood bushes I'd noted earlier in the day were loaded with berries as never before. It was going to be a cold, rough winter. Actually, it would probably be better to wait till the harsh weather passed and paint in the spring. I went over to the house to let Susan know we were getting an oscilloscope. ▼

*Well, it wasn't elementary, my dear Watson.*

# Make Me More Music,

by
Dorothy
Siegel

"Scott Joplin on Your Sci-Fi Hi-Fi" in last month's *ROM* described a method of generating musical tunes on a personal computer using fairly simple programs and a minimum of electronic hardware. Since that time, perhaps you've built and plugged in the interface unit we showed you how to make, input the BASIC and assembly language programs SCORE and PLAY, typed RUN, and heard your micro music machine output Scott Joplin. But after thirty or so reruns of "The Entertainer," by now you may be ready to add a little more variety to your repertoire.



## BASIC

"Bourree Anglaise" is from Bach's A Minor Unaccompanied Flute Sonata. Bach wrote eight flute sonatas, of which only the A Minor is for solo flute. This "sonata" is really a suite of four dance tunes, the last of which is called "Bouree Anglaise."

To hear this piece output by computer, you need only input the DATA statements supplied here in place of those for "The Entertainer" in the BASIC program SCORE given last month. Coding the music requires no alteration of SCORE aside from adjusting the tempo control in line 170 to K6 = .25.

```
899 REM “BOURREE ANGLAISE” FROM J.S.BACH’S A MINOR FLUTE SONATA
900 REM
901DATA“E 2E”,“A 2S”,“B 2S”,“C 2E”,“B 2S”,“A 2S”,“G#1E”
902DATA“A 2E”,“E 2E”,“E 2Q”
903DATA“E 1S”,“F 1S”,“E 1S”,“E 2S”,“E 1S”,“F 1S”,“E 1S”,“D 2S”
904DATA“E 1S”,“F 1S”,“E 1S”,“C 2S”,“B 2S”,“G#1S”,“E 2E”
905DATA“C 2S”,“A 2S”,“F 2E”,“D 2S”,“B 2S”,“G 2E”
906DATA“E 2S”,“C 2S”,“G 2E”,“G 2Q”
907DATA“E 2S”,“G 2S”,“E 2S”,“C 2S”,“G 1S”,“C 2S”,“E 2S”,“G 2S”
908DATA“D 2S”,“G 2S”,“D 2S”,“B 2S”,“G 1S”,“B 2S”,“D 2S”,“F 2S”
909DATA“E 2S”,“G 2S”,“E 2S”,“C 2S”,“G 1S”,“C 2S”,“E 2S”,“G 2S”
910DATA“D 2S”,“G 2S”,“D 2S”,“B 2S”,“G 1S”,“B 2S”,“D 2S”,“F 2S”
911DATA“E 1S”,“F 1S”,“G 1E”,“G 1S”,“A 2S”,“B 2E”
912DATA“C 2S”,“B 2S”,“C 2E”,“B!3Q”
913DATA“F 1S”,“C 2S”,“F 2E”,“A 3S”,“G 2S”,“A 3E”
914DATA“F#1S”,“A 2S”,“D 2E”,“C 3Q”
915DATA“B 3E”,“A 3S”,“G 2S”,“D 3E”,“F 2E”
916DATA“E 2S”,“D 2S”,“E 2E”,“C 3E”,“E 2E”
917DATA“F 2S”,“A 3S”,“F 2S”,“D 2S”,“D 2S”,“F 2S”,“D 2S”,“B 2S”
918DATA“B 2S”,“D 2S”,“B 2S”,“G 1S”,“G 2E”,“F 2E”
919DATA“E 2E”,“D 2S”,“C 2S”,“D 2E”,“B 2E”
920DATA“C 2Q.”,“G 2E”
921DATA“E 2S”,“D 2S”,“C 2E”,“C 2S”,“D 2S”,“E 2E”
922DATA“D 2S”,“C 2S”,“D 2E”,“B 3Q”
923DATA“G#1S”,“B 2S”,“D 2E”,“F 2E”,“E 2E”
924DATA“D 2S”,“C 2S”,“B 2S”,“C 2S”,“A 2E”,“C#2E”
925DATA“D 2S”,“E 2S”,“F 2E”,“E 2S”,“D 2S”,“C#2E”
926DATA“D 2S”,“A 2S”,“A 3E”,“A 3Q”
927DATA“C#2S”,“D 2S”,“E 2E”,“D 2S”,“C#2S”,“B 2E”
928DATA“C#2S”,“A 2S”,“G 2E”,“G 2Q”
```

Illustrated by Linda Smythe

# Maestro Micro

Almost any melody can be coded the way we coded "The Entertainer" and input as DATA statements in the BASIC program SCORE itemized last month. Then once the assembled version of the assembly language program PLAY is loaded at 0, or wherever you choose to load it, you have only to type RUN to see the coded music as numbers on your video screen and to hear it output as computerized music. A nice feature of the program is that any typing errors will appear on your screen as error messages and be heard as wrong notes. Let's try a few tunes.

## BACH

```
929DATA“F 2S”,“A 3S”,“F 2S”,“D 2S”,“A 2S”,“D 2S”,“F 2S”,“A 3S”
930DATA“E 2S”,“A 3S”,“E 2S”,“C#2S”,“A 2S”,“C#2S”,“E 2S”,“G 2S”
931DATA“F 2S”,“A 3S”,“F 2S”,“D 2S”,“A 2S”,“D 2S”,“F 2S”,“A 3S”
932DATA“E 2S”,“A 3S”,“E 2S”,“C#2S”,“A 2S”,“C#2S”,“E 2S”,“G 2S”
933DATA“F 2S”,“G 2S”,“A 3E”,“A 2E”,“D 2S”,“C#2S”
934DATA“D 2E”,“A 2E”,“D 1E.”,“F 2S”
935DATA“G 1S”,“A 2S”,“B 2E”,“B 2S”,“C 2S”,“D 2E”
936DATA“D 2S”,“E 2S”,“F 2E”,“F 2Q”
937DATA“G#1S”,“A 2S”,“B 2E”,“B 2S”,“C 2S”,“D 2E”
938DATA“D 2S”,“E 2S”,“F 2E”,“B 3E”,“E 2E”
939DATA“C 3S”,“B 3S”,“A 3S”,“G 2S”,“F#2S”,“E 2S”,“D#2S”,“E 2S”
940DATA“G 1S”,“B 3S”,“A 3S”,“G 2S”,“F#2S”,“E 2S”,“D#2S”,“E 2S”
941DATA“A 2S”,“C 3S”,“B 3S”,“A 3S”,“G 2S”,“F#2S”,“E 2S”,“D#2S”
942DATA“G 1S”,“B 3S”,“A 3S”,“G 2S”,“F#2S”,“E 2S”,“D#2S”,“E 2S”
943DATA“C 2S”,“B 2S”,“C 2E”,“A 3E”,“F#2E”
944DATA“D#2S”,“F#2S”,“B 2E”,“G 2E”,“E 1E”
945DATA“A 2S”,“G 2S”,“F#2E”,“B 2S”,“E 2S”,“D#2E”
946DATA“E 2E”,“B 2E”,“E 1E”,“E 2S”,“F 2S”
947DATA“G 2S”,“F 2S”,“G 2E”,“A 2S”,“C#2S”,“E 2E”
948DATA“G 2S”,“E 2S”,“F 2E”,“D 1E”,“D 2S”,“E 2S”
949DATA“F 2S”,“E 2S”,“F 2E”,“G 1S”,“B 2S”,“D 2E”
950DATA“F 2S”,“D 2S”,“E 2E”,“C 1E”,“A 3E”
951DATA“G#2E”,“G 2E”,“C#2E”,“G 2E”
952DATA“F#2E”,“F 2E”,“B 2E”,“F 2E”
953DATA“E 2E”,“F 2S”,“E 2S”,“D 2S”,“C 2S”,“B 2S”,“A 2S”
954DATA“G#1E”,“F#1S”,“G#1S”,“E 1E”,“E 2E”
955DATA“A 2S”,“B 2S”,“C 2E”,“B 2S”,“A 2S”,“G#1E”
956DATA“A 2E”,“E 2E”,“E 2Q”
957DATA“E 1S”,“F 1S”,“E 1S”,“E 2S”,“E 1S”,“F 1S”,“E 1S”,“D 2S”
958DATA“E 1S”,“F 1S”,“E 1S”,“C 2S”,“B 2S”,“G#1S”,“E 2E”
959DATA“C 2S”,“A 2S”,“F#2E”,“D 2S”,“B 2S”,“G#2E”
960DATA“E 2S”,“C 2S”,“A 3E”,“A 3E.”,“E 2S”
961DATA“D 2S”,“C 2S”,“B 2S”,“A 2S”,“E 1E”,“A 2S”,“G#1S”
962DATA“A 2S”,“C 2S”,“E 2E”,“E 2S”,“D#2S”,“E 2E”
963DATA“A 3E”,“D 2E”,“C#2E”,“G 2E”
964DATA“F#2E”,“C 2E”,“B 2E”,“F 2E”
965DATA“E 2E”,“F 2S”,“E 2S”,“D 2S”,“C 2S”,“B 2S”,“A 2S”
966DATA“G#1S”,“A 2S”,“B 2S”,“G#1S”,“E 1S”,“F#1S”,“G#1S”,“A 2S”
967DATA“B 2S”,“G#1S”,“B 2S”,“D 2S”,“D 2S”,“B 2S”,“D 2S”,“F 2S”
968DATA“F 2S”,“D 2S”,“F 2S”,“G#2S”,“G#2S”,“B 3S”,“E 2E”
969DATA“C 2E”,“B 2S”,“A 2S”,“C 2S”,“B 2S”,“A 2S”,“G#1S”
970DATA“A 2H”,“X”
```

# THE WALTZING COMPUTER

The music for Chopin's Waltz in D Flat, Op. 64, No. 1 seemed to me an excellent candidate for computerization. The great difficulties human fingers encounter trying to move swiftly over a piano keyboard do not exist for the computer, since its digits are not of the human variety. Change a few lines in the BASIC program as discussed here, input the new DATA statements, and you'll hear Chopin's famous "Minute (and a Half) Waltz."

Wondering how Chopin's eighth-note triplets and other musical ornaments would sound played a tempo with exact rhythmic precision, I added lines 465 and 475, obtaining the capability of specifying triplet quarter notes and triplet eighth notes by giving different values to T. But I didn't want the music to sound stiff and unmusical. For rhythmic flexibility I had to have more possible values for T, so I added lines 455, 463, and 467. Now I was able to code trills and grace notes and rubato passages. Adding lines 425 and 545 gave me a fourth octave and double dotted notes. I sped up the music (after listening for errors at a slower speed) by changing K6 in line 170. Then, to make the program run faster, I added line 155 so that 10 ^ 6 is calculated only once and inserted into line 570 instead of being calculated again and again for each note. Maybe you'll find other ways to speed up the program.

Now all that remained to be done was to decide how to code unusual rhythmic patterns, for instance, the four quarter notes in three-quarter time in measure 60, or line 1060, grace notes (beginning in lines 1069-1070), trills (lines 1081-88), and free runs (lines 1136-1140). These patterns I coded exactly as if I were going to play them on a standard musical instrument—with the help of some trial and error. For example, I tried using a thirty-second note trill in lines 1081-88; it sounded like a sick bird struggling to warble. A sixteenth note trill turned out to be more appropriate to the fast tempo.

Your musical judgment may differ from mine. You may decide to change the waltz's tempo, or the way the ornaments are coded, or how the last few bars are handled. Try different values for T and with dotted and double-dotted notes. By lengthening a note slightly, you can add stress to it and output more expressive music. Try adding more octaves. Try coding fancy turns and appoggiaturas. Transpose the music up (or down) by adding to (or subtracting from) P in line 435 for each semitone you wish to hear. Experimenting is what your micro music machine is for.

```
100 REM  TITLE \ "MINUTE (AND A HALF) WALTZ"
110 REM   THIS PROGRAM CALCULATES PITCH AND
120 REM   DURATION CONSTANTS FOR THE 8080
130 REM   ASSEMBLY LANGUAGE ROUTINE "PLAY".
140 REM
150 LET U=256 \ REM U DEFINES SCORE AREA IN MEMORY
155 LET Q1=10^6
160 LET K1=2^(1 / 12)
170 LET K6=.62\ REM TEMPO CONTROL
180   DIM Z$(5)
190 FOR V=1 TO 1000
200   LET C=1
210   READ Z$
220   LET N=100
230       IF Z$(1,1) = "A" THEN N=1
240       IF Z$(1,1) = "B" THEN N=3
250       IF Z$(1,1) = "C" THEN N=4
260       IF Z$(1,1) = "D" THEN N=6
270       IF Z$(1,1) = "E" THEN N=8
280       IF Z$(1,1) = "F" THEN N=9
290       IF Z$(1,1) = "G" THEN N=11
300       IF Z$(1,1) = "X" THEN GOTO 720
310        IF N=100 THEN GOTO 760
320   LET C=2
330   LET M=100
340       IF Z$(2,2) = "!" THEN M=N-1
350       IF Z$(2,2) = "#" THEN M=N+1
360       IF Z$(2,2) = " " THEN M=N
370        IF M=100 THEN GOTO 760
380   LET C=3
390   LET P=100
400       IF Z$(3,3) = "1" THEN P=M
410       IF Z$(3,3) = "2" THEN P=M+12
420       IF Z$(3,3) = "3" THEN P=M+24
425       IF Z$(3,3) = "4" THEN P=M+36
430        IF P=100 THEN GOTO 760
435        LET P=P+0 \ REM TRANSPOSE BY ADDING SEMITONES
440   LET C=4
450   LET T=100
455       IF Z$(4,4) = "C" THEN T=32
460       IF Z$(4,4) = "S" THEN T=16
463       IF Z$(4,4) = "F" THEN T=14
465       IF Z$(4,4) = "B" THEN T=12 \REM TRIPLET EIGHTHS
467       IF Z$(4,4) = "D" THEN T=10
470       IF Z$(4,4) = "E" THEN T=8
475       IF Z$(4,4) = "A" THEN T=6\REM TRIPLET QUARTERS
480       IF Z$(4,4) = "Q" THEN T=4
490       IF Z$(4,4) = "H" THEN T=2
500       IF Z$(4,4) = "W" THEN T=1
510       IF T=100 THEN GOTO 760
520        IF LEN (Z$) =4 THEN GOTO 560
530       LET C=5
540           IF Z$(5,5) = "." THEN T=2*T / 3
545           IF Z$(5,5) = ":" THEN T=4*T / 7
550   REM CALCULATE CONSTANTS
560       LET F1=220*(K1^(P-1))
570       LET T1=Q1 / (2*F1)
580       LET K3= (T1-24.5) / 7.5
590        LET K4=F1/ (K6*T)
600 LET D4=INT(K4) \ REM MAKE DURATION EVEN#
620 LET D5=INT(D4 / 256)\ REM CALC. 2 BYTES
630 LET D6=D5+1 \ REM D6=MSB
640 LET D7=D4+1-D5*256 \REM D7=LSB
645 REM THE +1'S ARE ADJUSTMENTS FOR "PLAY"
646 REM ROUTINE COUNTER.
650 REM TRANSFER CONSTANTS TO SCORE AREA.
660 FILL U+3*(V-1),INT(K3+.5)
670 FILL U+3*(V-1) +1,D7
680 FILL U+3*(V-1) +2,D6
690 PRINT V,
700 NEXT V
710 STOP
720 FILL U+3*(V-1),0
730 PRINT
740 PRINT "SCORE COMPILATION COMPLETE!"
745 B1=CALL(3) \ REM USE BASIC STACK
750 STOP
760 PRINT "ERROR IN NOTE #",V
770 PRINT "DATA STRING ",Z$
780 PRINT "CHARACTER #",C
790 STOP
800 END
810 REM
820 REM " THE MINUTE WALTZ" BY F. CHOPIN
```

```
821 REM
1001DATA “A!2Q”,“G 1E”,“A!2E”,“C 2E”,“B!2E”
1002DATA “G 1E”,“A!2E”,“B!2E”,“A!2E”,“C 2E”,“B!2E”
1003DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1004DATA “C 2E”,“B!2E”,“G 1E”,“A!2E”,“C 2E”,“B!2E”
1005DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1006DATA “C 2E”,“B!2E”,“G 1E”,“A!2E”,“C 2E”,“B!2E”
1007DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1008DATA “B!2E”,“C 2E”,“D!2E”,“E!2E”,“F 2E”,“G!2E”
1009DATA “B!3Q.”,“A!3E”,“G!2E”,“F 2E”
1010DATA “F 2E”,“E!2E”,“E!2C”,“F 2C”,“E!2S”,“D 2E”,“E!2Q”
1011DATA “B!3Q.”,“A!3E”,“G!2E”,“F 2E”
1012DATA “F 2E”,“E!2C”,“F 2C”,“E!2S”,“D 2E”,“E!2E”,“F 2E”,“B!2E”
1013DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1014DATA “C 2E”,“B!2E”,“G 1E”,“A!2E”,“C 2E”,“B!2E”
1015DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1016DATA “B!2E”,“C 2E”,“D!2E”,“E!2E”,“F 2E”,“G!2E”
1017DATA “B!3Q.”,“A!3E”,“G!2E”,“F 2E”
1018DATA “F 2E”,“E!2E”,“E!2C”,“F 2C”,“E!2S”,“D 2E”,“E!2Q”
1019DATA “B!3Q.”,“A!3E”,“G!2E”,“F 2E”
1020DATA “E!2E”,“F 2E”,“E!2C”,“F 2C”,“E!2S”,“D 2E”,“E!2E”,“E 2E”
1021DATA “F 2B”,“G!2B”,“F 2B”,“E 2E”,“F 2E”,“A!3E”,“G!2E”
1022DATA “F 2E”,“G!2E”,“F 2E”,“E 2E”,“F 2E”,“B!3E”
1023DATA “A!3B”,“B!3B”,“A!3B”,“G 2E”,“A!3E”,“C 3E”,“B!3E”
1024DATA “A!3E”,“B!3E”,“A!3E”,“G 2E”,“A!3E”,“D!3E”
1025DATA “C 3E”,“B!3E”,“A!3E”,“G!2E”,“F 2E”,“E!2E”
1026DATA “D!2E”,“C 2E”,“B!2E”,“A!2E”,“G!1E”,“F 1E”
1027DATA “E!1E”,“D!1E”,“C 1E”,“E!1E”,“B!2E”,“A!2E”
1028DATA “G 1E”,“A!2E”,“B!2E”,“C 2E”,“D!2E”,“E!2E”
1029DATA “F 2B”,“G!2B”,“F 2B”,“E 2E”,“F 2E”,“A!3E”,“G!2E”
1030DATA “F 2E”,“G!2E”,“F 2E”,“E 2E”,“F 2E”,“B!3E”
1031DATA “A!3B”,“B!3B”,“A!3B”,“G 2E”,“A!3E”,“C 3E”,“B!3E”
1032DATA “A!3E”,“B!3E”,“A!3E”,“G 2E”,“A!3E”,“F 3E”
1033DATA “E!3E”,“D!3E”,“C 3E”,“B!3E”,“A!3E”,“G!2E”
1034DATA “F 2E”,“E!2E”,“D!2E”,“C 2E”,“B!2E”,“A!2E”
1035DATA “A 2E”,“C 2E”,“B!2E”,“F 1E”,“G!1E”,“C 1E”
1036DATA “D!1H”,“F 2Q”
1037DATA “F 2B”,“G!2B”,“F 2B”,“E 2E”,“F 2E”,“A!3E”,“G!2E”
1038DATA “F 2E”,“G!2E”,“F 2E”,“E 2E”,“F 2E”,“B!3E”
1039DATA “A!3B”,“B!3B”,“A!3B”,“G 2E”,“A!3E”,“C 3E”,“B!3E”
1040DATA “A!3E”,“B!3E”,“A!3E”,“G 2E”,“A!3E”,“D!3E”
1041DATA “C 3E”,“B!3E”,“A!3E”,“G!2E”,“F 2E”,“E!2E”
1042DATA “D!2E”,“C 2E”,“B!2E”,“A!2E”,“G!1E”,“F 1E”
1043DATA “E!1E”,“D!1E”,“C 1E”,“E!1E”,“B!2E”,“A!2E”
1044DATA “G 1E”,“A!2E”,“B!2E”,“C 2E”,“D!2E”,“E!2E”
1045DATA “F 2B”,“G!2B”,“F 2B”,“E 2E”,“F 2E”,“A!3E”,“G!2E”
1046DATA “F 2E”,“G!2E”,“F 2E”,“E 2E”,“F 2E”,“B!3E”
1047DATA “A!3B”,“B!3B”,“A!3B”,“G 2E”,“A!3E”,“C 3E”,“B!3E”
1048DATA “A!3E”,“B!3E”,“A!3E”,“G 2E”,“A!3E”,“F 3E”
1049DATA “E!3E”,“D!3E”,“C 3E”,“B!3E”,“A!3E”,“G!2E”
1050DATA “F 2E”,“E!2E”,“D!2E”,“C 2E”,“B!2E”,“A!2E”
1051DATA “A 2E”,“C 2E”,“B!2E”,“F 1E”,“G!1E”,“C 1E”
1052DATA “D!1H”
1053DATA “A!2H.”,“E!1Q”
1054DATA “A!2H”,“E 1Q”
1055DATA “A!2H”,“F 1Q”
1056DATA “F 2H”
1057DATA “F 2H.”,“B!2Q”
1058DATA “F 2H”,“C 2Q”
1059DATA “E!2H”,“D!2Q”
1060DATA “C 2E.”,“E!2E.”,“D!2E.”,“B!2E.”
1061DATA “A!2H”,“E!1Q”
1062DATA “A!2H”,“E 1Q”
1063DATA “A!2H”,“F 1Q”
1064DATA “F 2H.”
1065DATA “C 2S”,“D!2S”,“C 2S”,“D!2S”,“B 2Q”,“C 2Q”
1066DATA “A!3Q”,“B!2Q”,“G 2Q”
1067DATA “A 2Q”,“G!2Q”,“A!2Q”
1068DATA “F 2Q”,“F 1Q”,“B!2Q”
1069DATA “A!2H”,“E!1E.”
1070DATA “A!3S”,“A!2H”,“E 1E.”
1071DATA “A!3S”,“A!2H”,“F 1E.”
1072DATA “A!3S”,“F 2H”,“F 2E.”
1073DATA “A!3S”,“F 2H”,“B!2E.”
1074DATA “A!3S”,“F 2H”,“C 2E.”
1075DATA “A!3S”,“E!2Q”,“D!2Q”,“C 2E.”
1076DATA “A!3S”,“E!2Q”,“D!2Q.”,“B!2S”
1077DATA “A!3S”,“A!2H”,“E!1E.”
1078DATA “A!3S”,“A!2H”,“E 1E.”
1079DATA “A!3S”,“A!2H”,“F 1Q”
1080DATA “F 2H.”,“F 2H”,“B!2Q”,“E!2H”,“A 2Q”,“E!2Q”,“A!2Q”,“D 2Q”
1081DATA “F 2Q”,“E!2Q”,“A!3Q”,“A!2S”,“B!2S”,“A!2S”,“B!2S”
1082DATA “A!2S”,“B!2S”,“A!2S”,“B!2S”,“A!2S”,“B!2S”,“A!2S”,“B!2S”
1083DATA “A!2S”,“B!2S”,“A!2S”,“B!2S”
1084DATA “A!2S”,“B!2S”,“A!2S”,“B!2S”,“A!2S”,“B!2S”,“A!2S”,“B!2S”
1085DATA “A!2S”,“B!2S”,“A!2S”,“B!2S”
1086DATA “A!2S”,“B!2S”,“A!2S”,“B!2S”,“A!2S”,“B!2S”,“A!2S”,“B!2S”
1087DATA “A!2S”,“B!2S”,“A!2S”,“B!2S”
1088DATA “A!2S”,“B!2S”,“A!2S”,“B!2S”,“A!2S”,“B!2S”,“A!2S”,“B!2S”
1089DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1090DATA “C 2E”,“B!2E”,“G 1E”,“A!2E”,“C 2E”,“B!2E”
1091DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1092DATA “C 2E”,“B!2E”,“G 1E”,“A!2E”,“C 2E”,“B!2E”
1093DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1094DATA “C 2E”,“B!2E”,“G 1E”,“A!2E”,“C 2E”,“B!2E”
1095DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1096DATA “B!2E”,“C 2E”,“D!2E”,“E!2E”,“F 2E”,“G!2E”
1097DATA “B!3Q.”,“A!3E”,“G!2E”,“F 2E”
1098DATA “F 2E”,“E!2E”,“E!2C”,“F 2C”,“E!2S”,“D 2E”,“E!2Q”
1099DATA “B!3Q.”,“A!3E”,“G2E”,“F 2E”
1100DATA “F 2E”,“E!2C”,“F 2C”,“E!2S”,“D 2E”,“E!2E”,“F 2E”,“B!2E”
1101DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1102DATA “C 2E”,“B!2E”,“G 1E”,“A!2E”,“C 2E”,“B!2E”
1103DATA “G 1E”,“A!2E”,“C 2E”,“B!2E”,“G 1E”,“A!2E”
1104DATA “B!2E”,“C 2E”,“D!2E”,“E!2E”,“F 2E”,“G!2E”
1105DATA “B!3Q.”,“A!3E”,“G!2E”,“F 2E”
1106DATA “F 2E”,“E!2E”,“E!2C”,“F 2C”,“E!2S”,“D 2E”,“E!2Q”
1107DATA “B!3Q.”,“A!3E”,“G!2E”,“F 2E”
1108DATA “E!2E”,“F 2E”,“E!2C”,“F 2C”,“E!2S”,“D 2E”,“E!2E”,“E 2E”
1109DATA “F 2B”,“G!2B”,“F 2B”,“E 2E”,“F 2E”,“A!3E”,“G!2E”
1110DATA “F 2E”,“G!2E”,“F 2E”,“E 2E”,“F 2E”,“B!3E”
1111DATA “A!3B”,“B!3B”,“A!3B”,“G 2E”,“A!3E”,“C 3E”,“B!3E”
1112DATA “A!3E”,“B!3E”,“A!3E”,“G 2E”,“A!3E”,“D!3E”
1113DATA “C 3E”,“B!3E”,“A!3E”,“G!2E”,“F 2E”,“E!2E”
1114DATA “D!2E”,“C 2E”,“B!2E”,“A!2E”,“G!1E”,“F 1E”
1115DATA “E!1E”,“D!1E”,“C 1E”,“E!1E”,“B!2E”,“A!2E”
1116DATA “G 1E”,“A!2E”,“B!2E”,“C 2E”,“D!2E”,“E!2E”
1117DATA “F 2B”,“G!2B”,“F 2B”,“E 2E”,“F 2E”,“A!3E”,“G!2E”
1118DATA “F 2E”,“G!2E”,“F 2E”,“E 2E”,“F 2E”,“B!3E”
1119DATA “A!3B”,“B!3B”,“A!3B”,“G 2E”,“A!3E”,“C 3E”,“B!3E”
1120DATA “A!3E”,“B!3E”,“A!3E”,“G 2E”,“A!3E”,“F 3E”
1121DATA “E!3E”,“D!3E”,“C 3E”,“B!3E”,“A!3E”,“G!2E”
1122DATA “F 2E”,“E!2E”,“D!2E”,“C 2E”,“B!2E”,“A!2E”
1123DATA “A 2E”,“C 2E”,“B!2E”,“F 1E”,“G!1E”,“C 1E”
1124DATA “D!1H”,“F 2Q”
1125DATA “F 2B”,“G!2B”,“F 2B”,“E 2E”,“F 2E”,“A!3E”,“G!2E”
1126DATA “F 2E”,“G!2E”,“F 2E”,“E 2E”,“F 2E”,“B!3E”
1127DATA “A!3B”,“B!3B”,“A!3B”,“G 2E”,“A!3E”,“C 3E”,“B!3E”
1128DATA “A!3E”,“B!3E”,“A!3E”,“G 2E”,“A!3E”,“D!3E”
1129DATA “C 3E”,“B!3E”,“A!3E”,“G!2E”,“F 2E”,“E!2E”
1130DATA “D!2E”,“C 2E”,“B!2E”,“A!2E”,“G!1E”,“F 1E”
1131DATA “E!1E”,“D!1E”,“C 1E”,“E!1E”,“B!2E”,“A!2E”
1132DATA “G 1E”,“A!2E”,“B!2E”,“C 2E”,“D!2E”,“E!2E”
1133DATA “F 2B”,“G!2B”,“F 2B”,“E 2E”,“F 2E”,“A!3E”,“G!2E”
1134DATA “F 2E”,“G!2E”,“F 2E”,“E 2E”,“F 2E”,“B!3E”
1135DATA “A!3B”,“B!3B”,“A!3B”,“G 2E”,“A!3E”,“C 3E”,“B!3E”
1136DATA “A!3E”,“B!3E”,“A!3E”,“G 2E”,“A!3E”,“F 4E”
1137DATA “E!4D”,“D!4D”,“C 4D”,“B!4D”,“A!4D”,“G!3D”
1138DATA “F 3D”,“E!3B”,“D!3B”,“C 3B”,“B!3B”,“A!3B”,“G!2B”,“F 2B”
1139DATA “E!2F”,“D!2F”,“C 2F”,“B!2F”,“A 2D”
1140DATA “C 2D”,“B!2D”,“F 1A”,“G!1A”,“C 1Q”,“D!1H”
9999DATA “X”
```

# DUET FOR PERSON AND MICROCOMPUTER

Something that really caught my fancy was the idea of playing duets with my computer. I set out to find an appropriate piece of music to try, perferably a fairly simple one that would not overly strain my limited technique. Bach's Invention No. 4 in D Minor, one of thirty two-part and three-part Inventions written around 1723, seemed perfect.

I made an arrangement that would fit the frequency range of most musical instruments. Then I coded one part for the computer. Using the BASIC program as altered for the Chopin waltz, I input new DATA statements for the Invention. Listening to the output carefully, I corrected wrong notes, changed the tempo (line 170), and adjusted the pitch (line 560) to match my flute's. (If I had needed to change key to match a transposing instrument, like an E saxophone, for instance, I would have altered line 435 by adding 3 to P.) I typed RUN, watched the notes crunch through SCORE, waited to hear the first two measures of music, and started playing along—in perfect time, of course—with the computer.

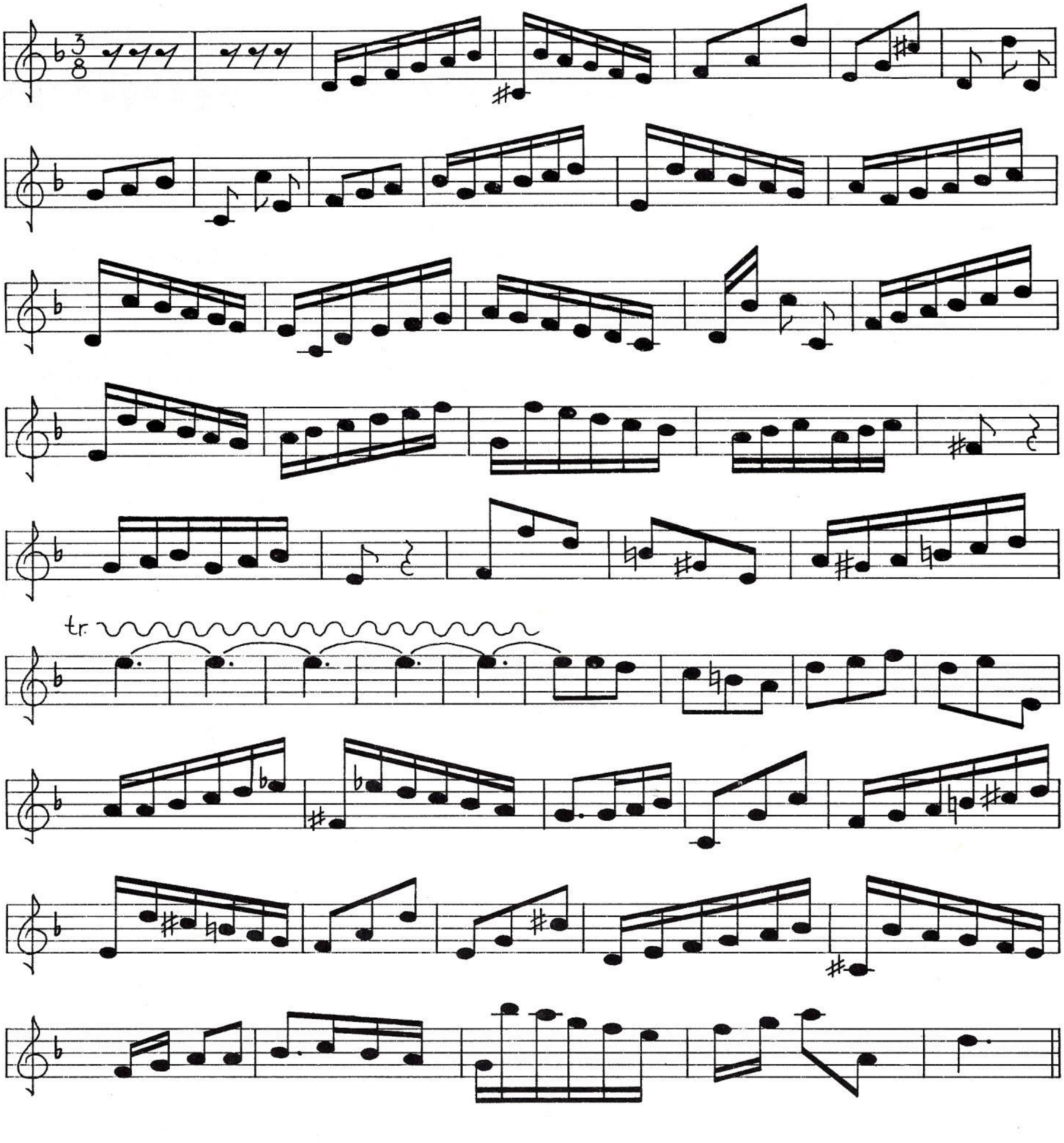
In all my duet playing, I've never found a partner so amenable to playing his/her/its part *my* way.

```
820 REM "INVENTION #4 IN D MINOR" BY J.S.BACH
821 REM
901DATA"D 1S","E 1S","F 1S","G 1S","A 2S","B!2S"
902DATA"C#1S","B!2S","A 2S","G 1S","F 1S","E 1S"
903DATA"F 1E","A 2E","D 2E"
904DATA"G 1E","C#2E","E 2E"
905DATA"D 2S","E 2S","F 2S","G 2S","A 3S","B!3S"
906DATA"C#2S","B!3S","A 3S","G 2S","F 2S","E 2S"
907DATA"F 2S","D 2S","E 2S","F 2S","G 2S","A 3S"
908DATA"B!2S","A 3S","G 2S","F 2S","E 2S","D 2S"
909DATA"E 2S","C 2S","D 2S","E 2S","F 2S","G 2S"
910DATA"A 2S","G 2S","F 2S","E 2S","D 2S","C 2S"
911DATA"D 2S","E 2S","F 2S","D 2S","E 2S","F 2S"
912DATA"G 1Q."
913DATA"C 2S","D 2S","E 2S","C 2S","D 2S","E 2S"
914DATA"F 1Q","B!2Q"
915DATA"A 2E","G 1E"
916DATA"C 2S","B!2S","A 2S","G 1S","F 1S","E 1S"
917DATA"F 1S","G 1S","G 1E.","F 1S"
918DATA"F 1E","C 2E","C 2E"
919DATA"C 2C","D 2C","C 2C","D 2C","C 2C","D 2C"
920DATA"C 2C","D 2C","C 2C","D 2C","C 2C","D 2C"
921DATA"C 2C","D 2C","C 2C","D 2C","C 2C","D 2C"
922DATA"C 2C","D 2C","C 2C","D 2C","C 2C","D 2C"
923DATA"C 2C","D 2C","C 2C","D 2C","C 2C","D 2C"
924DATA"C 2C","D 2C","C 2C","D 2C","C 2C","D 2C"
932DATA"C 2S","B!2S","A 2S","G 1S","F 1S","E 1S"
933DATA"C 2S","D 1S","E 1S","F#1S","G 1S","A 2S"
934DATA"B!2S","A 2S","G 1S","F 1S","E 1S","D 1S"
935DATA"B!2S","C 1S","D 1S","E 1S","F 1S","G 1S"
936DATA"A 2S","B 2S","C 2S","D 2S","E 2S","F 2S"
937DATA"G#1S","F 2S","E 2S","D 2S","C 2S","B 2S"
938DATA"C 2S","B 2S","D 2S","C 2S","B 2S","A 2S"
939DATA"G#1S","A 2S","G#1S","F#1S","E 1S","D 1S"
940DATA"C 1S","D 1S","E 1S","F#1S","G#1S","A 2S"
941DATA"D 1S","C 2S","B 2S","A 2S","G#1S","F#1S"
942DATA"E 1S","F#1S","G#1S","A 2S","B 2S","C 2S"
943DATA"F#1S","E 2S","D 2S","C 2S","B 2S","A 2S"
944DATA"G#1S","A 2S","B 2S","C 2S","D 2S","E 2S"
945DATA"A 2S","F 2S","E 2S","D 2S","C 2S","B 2S"
946DATA"A 3S","G#2S","F#2S","E 2S","A 3E."
947DATA"D 2S","B 2C","C 2C","B 2E","A 2S"
948DATA"A 2E.","A 2S","B!2S","C 2S"
949DATA"D 1E","F#1E","A 2E"
950DATA"B!2S","G 1S","A 2S","B!2S","C 2S","D 2S"
951DATA"E 1S","D 2S","C 2S","B!2S","A 2S","G 1S"
952DATA"A 2E","F 2S","E 2S","F 2E"
953DATA"G 1E","E 2Q"
954DATA"D 2S","E 2S","F 2S","G 2S","A 3S","B!3S"
955DATA"C#2S","B!3S","A 3S","G 2S","F 2S","E 2S"
956DATA"F 2E","D 2E","G 1E"
957DATA"D 2S","C#2S","E 2S","A 2S","C#2S"
958DATA"D 2S","B 2S","C#2E.","D 2S"
959DATA"D 2S","C 2S","B!2S","A 2S","G 1S","F 1S"
960DATA"B!2S","C#1S","D 1S","E 1S","F 1S","G 1S"
961DATA"A 2S","D 2F","F 1A","E 1D","D 1D"
962DATA"D 1H","X"
```

There are a lot of things you can do with the PLAY program. You can arrange for musical rests, notes lower than middle C (262Hz), sweeps up and down the frequency spectrum. . . . You can tune the computer by slightly increasing or decreasing the 220 in line 560 to raise or lower the frequency of the music. Try getting quarter-tone music by changing the 12 in line 160 to a 24. Arrange clicks and/or ticks for use as a metronome. If

# Invention No. 4 in D Minor

**J. S. Bach**
**(Arr. D. Siegel)**



you are practicing a difficult musical phrase, work it out on the computer first so that you can actually hear what the music is supposed to sound like; then play it back on the computer slowly as you work on the phrase, imitating the computer, gradually increasing the speed until you're playing it up to tempo.

There are limits to what our humble programs and interface can do, obviously; the lack of amplitude and envelope control is a real drawback, for instance. But if you've got the computer and you've got a musical instrument, why not put the two together?

One warning: you may find yourself—or your computer—hooked enough on micro-generated music to add a digital-to-analog converter, plus more hardware, plus more complex software, so the two of you can graduate to more imaginative sci-fi hi-fi outputting. ▼

# WINGS

# Computer Models

## by Joseph Weizenbaum

Suppose a team of explorers from a highly technological society just like ours, but one that knew nothing about computers, were to come upon a functioning computer. They find that they cannot break into it, can gain no access to its, so to say, electroneurophysiological apparatus. They do notice, however, that whenever they type something on its console typewriter, the computer's lights flash in a complex but apparently orderly way, its magnetic tapes sometimes spin, and the typewriter types a message that appears to be a response to what they have typed. After a time, they discover that they can dismount the computer's magnetic tapes and cause their contents to be printed on another device, a high-speed printer, which they also find on the site of their exploration. These contents prove to be readable, at least in the sense that they are represented in the explorer's own alphabet.

Since this machine—and the explorers do recognize it as a machine—is obviously a behaving instrument, the explorers naturally wish to discover the laws of its behavior. How could they go about reaching the understanding they desire? Indeed, what can it mean to understand the machine's laws of behavior?

We, of course, can put ourselves in the position of a highly privileged observer, somewhat like that of a chemistry instructor who knows very well what compound he gave his students to analyze. We know that the machine the explorers found is a computer, moreover, a computer of precisely such and such a type and one containing a particular program we also know in detail. We can therefore tell the precise degree, so to say, of understanding the explorers will have achieved at any given stage of their research. If, for example, they were to produce a computer of their own which, as seen from our privileged perspective, appears to be an exact copy of the computer they found and which even contains the same program as the original, then we would have to say that they understood the original computer at least as well as did its designers.

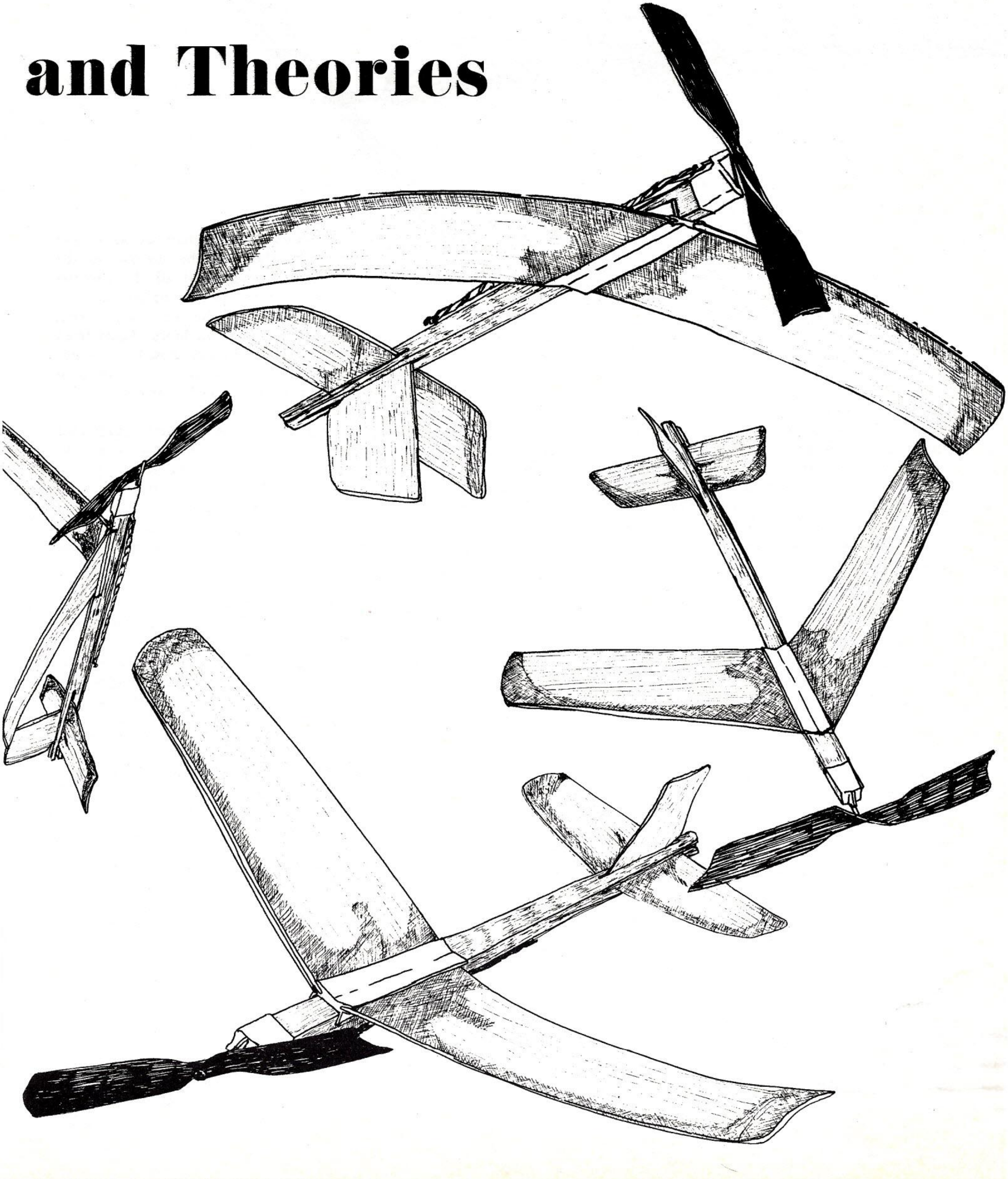
However, lesser achievements would also deserve to be called understanding of a very high degree. Suppose, for example, that the explorers managed to build a computer whose internal structure and whose internal components are entirely different, but whose input-output behavior is indistinguishable from that of the original; in other words, no test short of breaking open either computer can determine which of the two computers, the one the explorers found or the one they built, generated what response to what input. It may be that the internal components of the found machine are made of bailing wire, chewing gum, and adhesive tape, whereas those of the explorers' functional copy are all electronic; that doesn't matter as long as, for any reason, the original machine may not be opened for detailed internal inspection. (Actually, of course, such an achievement is impossible in principle. It may be, for example, that the original computer was so constructed that it prefaces its first console typewriter output with an exclamation mark on and only on the seventeenth Thursday of every leap year. Even if that were discovered, the explorers could never be sure that there are not other oddities which, though systematic, have not yet been discovered. We, as privileged observers, would, of course, know about such things.)

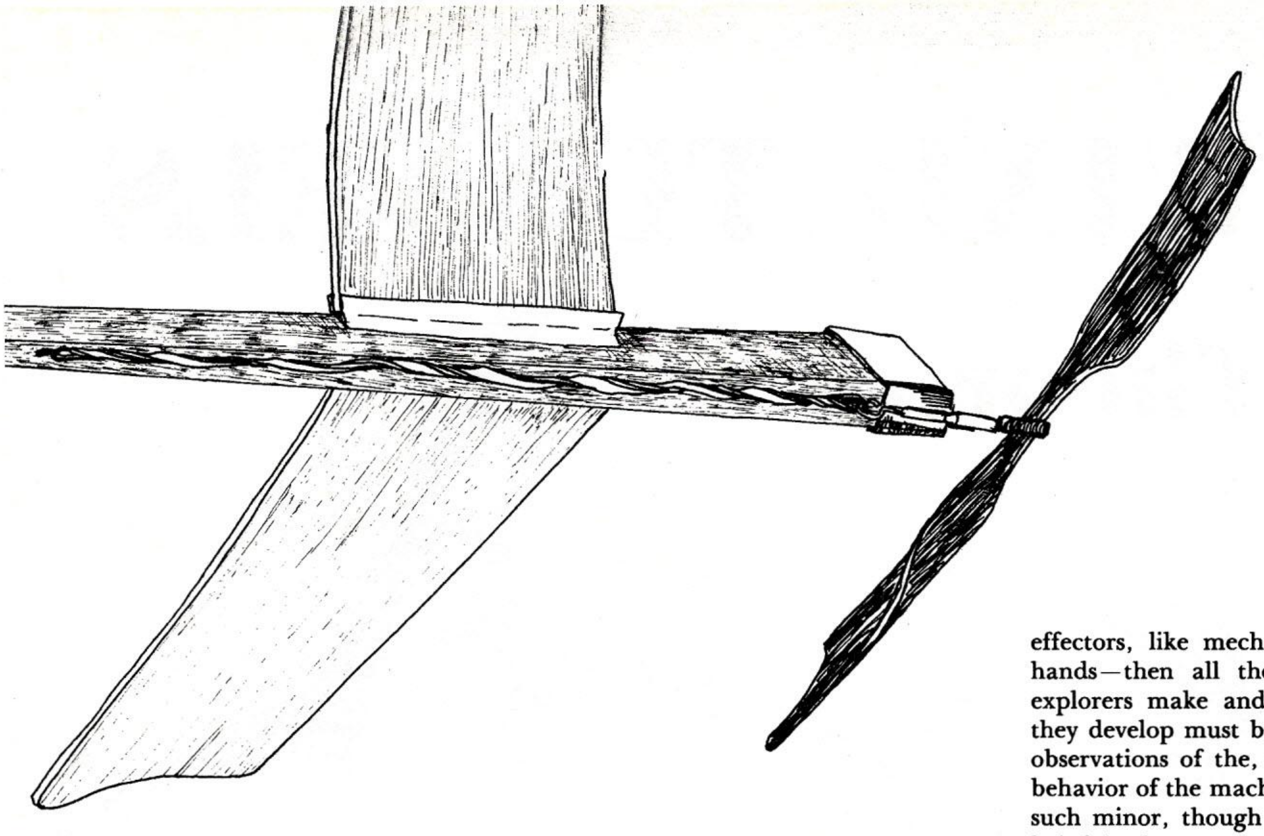
A still lesser degree of understanding could be claimed if the explorers succeeded in building some sort of digital computer, say, a simple universal Turing machine of the type we discussed in the August issue of *ROM,* and then explained the machine they found in terms of Turing-machine principles. They could then account for the found machine's extraordinary versatility and even for the fact, say, that it takes it longer to compute the inverse of a large matrix than that of a smaller one. They might, on the other hand, be utterly unable to explain why it takes the found computer longer to execute algorithms given to it in one programming language than it does to execute those same algorithms written in another programming language. We, given our omniscience about computers, know that the difference in execution speeds is due to the fact that the computer translates programs in the first language a line at a time into its machine language, and then obeys the so-generated machine-language instructions, whereas it first translates the whole program written in the second language into its machine language and only then executes the entire set of so-generated machine-

From *Computer Power and Human Reason,* Copyright © 1976 by W. H. Freeman and Co.

Illustrated by Cindy Hain

# IN WIND TUNNELS and Theories





language instructions. The latter process is almost always much less time-consuming than the former. Presumably the explorers would eventually develop some explanation for this and other puzzling phenomena. They might, for example, conjecture that some programming languages are more familiar to the machine than some others, and might even develop some taxonomy of programming languages based on the machine's experimentally discovered familiarity with them. The concept of familiarity, as well as the taxonomy of languages for which it serves as an organizing principle, would then become part of their computer science. It is, of course, a concept weak in explanatory power, even a misleading one. But then it is much easier for us privileged observers to know this than it is for the explorers, faced as they are with the task of having to explain phenomena of unbounded complexity.

Let us press this fantasy one more step: Suppose the explorers found not just one computer, but many computers of many diverse types, all of them, however, so-called single-address machines. Recall that a single-address machine is one whose built-in instructions have the form "operation code; address of datum to be operated on". With luck, and if the explorers were clever, they would discover what they would undoubtedly call a "language universal" with respect to the grammatical structure of the machine languages they have encountered. And to explain it as something other than a mere accident (which would, of course, be no explanation at all), they would have to conclude that this universal feature of all the languages they have observed must be due to some correspondingly universal feature, some innate property, of the machines themselves. And they would, as we privileged observers know, of course, be correct; the fact that a computer's machine language has the single-address format is a direct consequence of its design. Indeed, if we assume that the machines the explorers discovered are ordinary computers and not robots—that is, that they don't have perceptors, like television eyes, and effectors, like mechanical arms and hands—then all the discoveries the explorers make and all the theories they develop must be based solely on observations of the, so to say, verbal behavior of the machines. Apart from such minor, though possibly not unhelpful, phenomena as the flashing of the computers' lights and the occasional motions of their tape reels, the only evidence of their structures that the computers provide is, after all, linguistic. They accept strings of linguistic inputs in the form of the texts typed on their console typewriters, and they respond with linguistic outputs written on the same instrument or onto magnetic tapes.

*If we may not break the computer open, then all our explanations must be linguistic ones.*

In the August and September issues of *ROM* we were very much concerned with legal moves in abstract games and grammatical constructions in abstract languages. My aim there was to build up the idea of a computer on the basis of such concepts. In the fantasy we are currently entertaining, we are, in effect, looking at the other side of the same coin. We now see that, if we strive to explain computers when bounded by the restriction that we may not break the computer open, then all explanations must be derived from linguistic bases.

The position of a human being observing another human being is not so very different from that of the explorers who wish to understand the computers they have encountered. We too have extremely limitied access to the neurophysiological material that appears to determine how we think. Besides, it wouldn't advance our current understanding of thinking very much even if we could subject the living brain to the kind of analysis to which we actually can subject a running computer, that is, by tracing connections, electrical pulses, and so on. Our ignorance of brain function is currently so very nearly total that we could not even begin to frame appropriate research strategies. We would stand before the open brain, fancy instruments in hand, roughly as an unschooled laborer might stand before the exposed wiring of a computer: awed perhaps, but surely helpless. A microanalysis of brain functions is, moreover, no more useful for understanding anything about thinking than a corresponding analysis of the pulses flowing through a computer would be for understanding what program the computer is running. Such analyses would simply be at the wrong conceptual level. They might help to decide crucial experiments, but only after such experiments had been designed on the basis of much higher-level (for example, linguistic) theories.

*Maybe the computer prefaces its output with an exclamation mark on and only on the seventeenth Thursday of a leap year.*

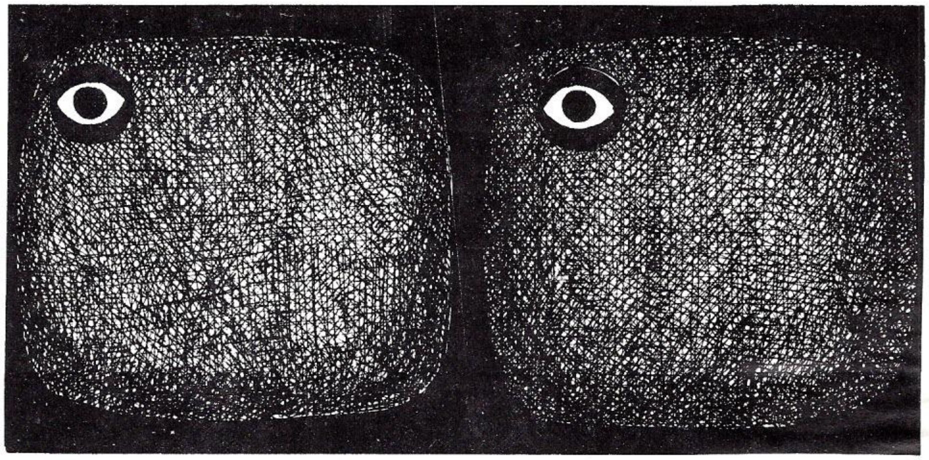
Because, in fact, scientists do suffer from the same sort of handicaps as we imposed on our mythical explorers, and cannot communicate with an omniscient observer who could, if he but would, reveal all secrets, it is not surprising that at least some scientists seek understanding of the way humans work in somewhat the same way as our explorers might have sought to understand the computers they found, that is, by designing computers whose input-output behavior resembles that of humans as closely as possible.

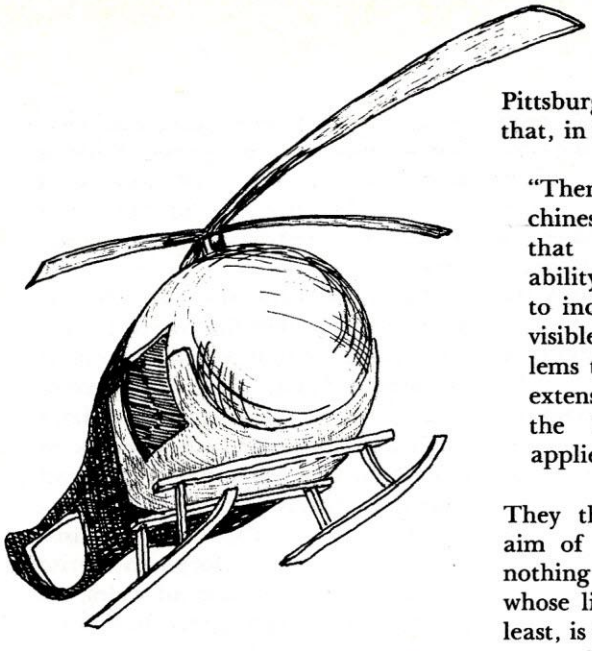
*We would stand before the open brain, fancy instruments in hand, awed perhaps, but surely helpless.*

The work of linguists—for example, that of Noam Chomsky—should be mentioned here, even though it does not involve the use of computers. A simple-minded and grossly misleading view of the task that Chomsky's school has set itself is that it is to systematically record the grammatical rules of as many natural languages (e.g., English) as possible. If that were the only, or even the principal, aim of Chomsky's school we would expect it to publish a series of books, all independent of one another, entitled *The Grammar of X,* where *X* stands for one of the various known human languages. In fact, Chomsky's most profoundly significant working hypothesis is that man's genetic endowment gives him a set of highly specialized abilities and imposes on him a corresponding set of restrictions which, taken together, determine the number and the kinds of degrees of freedom that govern and delimit all human language development.

To understand how a "specialized ability" may simultaneously be a "corresponding restriction," we need only remind ourselves of a single-address machine. The fact that such a machine can decode a machine-language instruction in terms of a component (say, its eight leftmost bits) that is the instruction's operation code, and another component (its remaining bits) that is its address portion, implies at once that no other instruction format is, for that machine, admissible. Indeed, the very idea of the grammaticality of a whole computer program, let alone that of a single machine-language instruction, implies that there exist some symbol strings which, while they may superficially look like programs, are unintelligible and hence not admissible as programs. What is seen from one point of view as a specialized ability of a machine must be seen as a restriction from another perspective.

How then, Chomsky asks, can we gain an insight into the genetically endowed abilities that we call the mind? He answers that, given our present state of virtually total ignorance about the living brain, our best chance is to infer the innate properties of the mind from the highly restrictive principles of a "universal grammar." The linguist's first task is therefore to write grammars, that is, sets of rules, of particular languages, grammars capable of characterizing all and only the grammatically admissible sentences of those languages, and then to postulate principles from which crucial features of all such grammars can be deduced. That set of principles would then constitute a universal grammar. Chomsky's hypothesis is, to put it another way, that the rules of such a universal grammar would constitute a kind of projective description of important aspects of the human mind. He does not believe, of course, that people know these rules in the same way that they know, say, the rules of long division. Instead they know them (to use Polanyi's word) tacitly, that is, in the same way that people know how to maintain their balance while running. In both





*Whatever else man is, and he is very much else, he is also an information processor.*

speaking and running, by the way, performance once mastered, deteriorates when an attempt is made to apply explicit rules consciously.

In an important sense, then, Chomsky is one of our mythical explorers. Unable to inspect the insides of the found objects, human minds, and ignorant of whatever engineering principles may be relevant, e.g., the neurophysiology of the living brain, he sets out to infer the found object's laws from the evidence of its linguistic behavior.

*AI will have to build machines that can feel, taste, see, and hear.*

As far-reaching as the research aims of Chomsky's school are, they are modest compared to those of the leading scientists working in that branch of computer science called "artificial intelligence" (AI). Herbert A. Simon and Allen Newell, for example, together leaders of one of the most productive teams of AI researchers at Carnegie-Mellon University, Pittsburgh, claimed as early as 1958 that, in their own words:

> "There are now in the world machines that think, that learn and that create. Moreover, their ability to do these things is going to increase rapidly until—in the visible future—the range of problems they can handle will be coextensive with the range to which the human mind has been applied."

They thus proclaimed the research aim of the new science, AI, to be nothing less than to build a machine whose linguistic behavior, to say the least, is to be equivalent to that of humans. Should AI realize this aim, it will have achieved the second, and very high indeed, level of understanding of human functions that we discussed for our explorers' understanding of the functions of the machines they encountered. In that context we fantasized that the explorers had succeeded in building a machine whose input-output behavior was, under any test whatever, indistinguishable from that of the machines they found, although the components of the two machine types need not have been the same.

In fact, the research goals of AI are much more ambitious than were those of our explorers, who intended only to understand how the machine they found generated its textual responses to the textual inputs it was given, whereas the goal of AI is to understand how an organism handles "a range of problems. . . coextensive with the range to which the human mind has been applied." Since the human mind has applied itself to, for example, problems of aesthetics involving touch, taste, vision, and hearing, AI will have to build machines that can feel, taste, see, and hear. Since the future in which machine thinking will range as widely as Simon and Newell claim it will is, at this writing, merely "visible" but not yet here, it is perhaps too early to speculate what sort of equipment machines will have to have in order to think about such human concerns as, say, disappointment in adolescent love. But there are machines today, principally at M.I.T., at Stanford University, and at the Stanford Research Institute, that have arms and hands whose movements are observed and coordinated by computer-controlled television eyes. Their hands have fingers which are equipped with pressure-sensitive pads to give them a sense of touch. And there are hundreds of machines that do routine (and even not so routine) chemical analyses, and that may therefore be said to have senses of taste. Machine production of fairly high-quality humanlike speech has been acheived, principally at M.I.T. and at the Bell Telephone Laboratories. The U.S. Department of Defense and the National Science Foundation are currently supporting considerable efforts toward the realization of machines that can understand human speech. Clearly, Simon's and Newell's ambition is taken seriously both by powerful U.S. government agencies and by a significant sector of the scientific community.

Given that individuals differ in their visual acuity, it is not to be expected that everyone even now can see the same future that was already visible to Simon and Newell in 1958. Nor is it necessary for psychologists to recognize the power of computer models of human functions in order to share Simon's and Newell's grandiose vision. Much humbler signs point the way, and even more directly.

Whatever else man is, and he is very much else, he is also a behaving organism. If man's understanding of himself is to be at least in part scientific, then science must be allowed to assume that at least some aspects of man's behavior obey laws that science can discover and formalize within some scientific conceptual framework. However naive and informal or, on the other hand, sophisticated and formal a notion of "information" one has in mind, it must be granted that man acts on (that is, responds to) information that impinges on him from his environment, and that his actions, especially his verbal behavior, inform his environment in turn. Whatever else man is, then, and again he is very much else, he is also a receiver and a transmitter of information. But even so, he is certainly more than a mere mirror that reflects more or less precisely whatever signals impinge on it; for he attends to only a small fraction of what William James called "the blooming, buzzing confusion" of sensations with which his environment bombards him, and he transforms that distillate of his world into memories, mental imagery of many sorts, speech and writing, strokes on piano keyboards, in short, into thought and behavior. Whatever else man is, then, and he is much else, he is also an information processor.

*Seeing man as an information-processing system does not in itself dehumanize him.*

I will, in what follows, try to maintain the position that there is nothing wrong with viewing man as an information processor (or indeed as anything else) nor with attempting to understand him from that perspective, providing, however, that we never act as though any single perspective can comprehend the whole man. Seeing man as an information-processing system does not in itself dehumanize him, and may very well contribute to his humanity in that it may lead him to a deeper understanding of one specific aspect of his human nature. It could, for example, be enormously important for man's understanding his spirituality to know the limits of the explanatory power of an information-processing theory of man. In order for us to know those limits, the theory would, of course, have to be worked out in considerable detail.

Before we discuss what an information-processing theory of man might look like, I must say more about theories and especially about their relation to models. A theory is first of all a text, hence a concatenation of the symbols of some alphabet. But it is a symbolic construction in a deeper sense as well; the very terms that a theory employs are symbols which, to paraphrase Abraham Kaplan, grope for their denotation in the real world or else cease to be symbolic. The words "grope for" are Kaplan's, and are a happy choice—for to say that symbols "find" their denotation in the real world would deny, or at least obscure, the fact that the symbolic terms of a theory can never be finally grounded in reality.

Definitions that define words in terms of other words leave those other words to be defined. In science generally, symbols are often defined in terms of operations. In physics, for example, mass is, informally speaking, that property of an object which determines its motion during collision with other objects. (If two objects moving at identical velocities come to rest when brought into head-on collision, it is said that they have the same mass.) This definition of mass permits us to design experiments involving certain operations whose outcomes "measure" the mass of objects. Momentum is defined as the product of the mass of an object and its velocity ($mv$), acceleration as the rate of change of velocity with time ($a = dv/dt$), and finally force as the product of mass and acceleration ($f = ma$). In a way it is wrong to say that force is "defined" by the equation $f = ma$. A more suitable definition given in some physics texts is that force is any influence capable of producing a change in the motion of a body. The difference between the two senses of "definition" alluded to here illustrates that so-called operational definitions of a theory's terms provide a basis for the design of experiments and the discovery of general laws, but that these laws may then serve as implicit definitions of the terms occuring in them. These and still other problematic aspects of definition imply that all theoretic terms, hence all theories, must always be characterized by a certain openness. No term of a theory can ever be fully and finally understood. Indeed, to once more paraphrase Kaplan, it may not be possible to fix the content of a single concept or term in a sufficiently rich theory (about, say, human cognition) without assessing the truth of the whole theory. This fact is of the greatest importance for any assessment of computer models of complex phenomena.

*The terms of a theory are symbols which grope for denotation in the real world.*

A theory is, of course, not merely any grammatically correct text that uses a set of terms somehow symbolically related to reality. It is a systematic aggregate of statements of laws. Its content, its very value as theory, lies at least as much in the structure of the interconnections that relate its laws to one another, as in the laws themselves. (Students sometimes prepare themselves for examinations in physics by memorizing lists of equations. They may well pass their examinations with the aid of such feats of memory, but it can hardly be said that they know physics, that, in other words, they command a theory.) A theory, at least a good one, is thus not merely a kind of data bank in which one can "look up" what would happen under such and such conditions. It is rather more like a map (an analogy Kaplan also makes) of a partially explored territory. Its function is often heuristic, that is, to guide the explorer in further discovery. The way theories make a difference in the world is thus not that they answer questions, but that they guide and stimulate intelligent search. And (again) there is no single "correct" map of a territory. An aerial photograph of an area serves a different heuristic function, say, for a land-use planner, than does a demographic map of the same area. One use of a theory, then, is that it prepares the conceptual categories within which the theoretician and the practitioner will ask his questions and design his experiments.

It must not be thought that this heuristic function of theory is manifest only in science. To name but one of the possible examples outside the sciences, Steven Marcus, the American literary critic, used theories of literary criticism freshly honed on the stone of psychoanalytic theory to do an essentially anthropological study of that "foreign, distinct, and exotic" subculture that was the sexual subculture of Victorian England. See his *The Other Victorians* (New York: Basic Books, 1966). More recently he wrote in the preface of his *Engels, Manchester, and the Working Class* (New York: Random House, 1974), "The present work may be regarded as part of a continuing experiment. . . to ascertain how far literary criticism can help us to understand history and society; to see how far the intellectual discipline that begins with the work of close textual analysis can help us understand certain social, historical, or theoretical documents." In neither book was a theory of literary criticism "applied," as, for example, a chemical theory may be applied to the chemical analysis of a compound; instead, Marcus' theories were used heuristically, as travelers use maps to explore strange territory.

Ordinarily, of course, when we speak of putting a theory to work, we mean drawing some consequences from it. And by that, in turn, we mean postulating some set of circumstances that involves some terms of the theory, and then asking what the theory says those particular circumstances imply for others of the theory's terms. We may describe the state of the economy of a specific country to an economist, for example, by giving him a set of the sorts of economic indices his particular economic theory accommodates. He may ask us some questions which, he would say, emerge directly from his theory. Such questions, by the way, might give us more insight into whether he is, say, a Marxist or a Keynesian economist than any answers he might ultimately give us, for they would reveal the structure of his theory, the network of connections between the economic laws in which he believes. Finally, we expect to be told what his theory says, e.g., that the country will do well, or that there will be a depression. More technically speaking, we may say that to put a theory to work means to assign specific values, by no means always numerical, to some of its parameters (that is, to the entities its terms signify), and then to methodically determine what values the theory assigns to other of its parameters. Often, of course, we arrive at the specifications to which we wish to apply a theory by interrogating or measuring some aspect of the real world. The input, so to speak, to a political theory may, for example, have been derived from public-opinion polls. At other times our specifications may be entirely hypothetical, as, for example, when we ask of physics what effect a long journey near the speed of light would have on the timekeeping property of a clock. In any case, we identify certain terms of the theory with what we understand them to denote, associate specifications with them, and, in effect, ask the theory to figure out the consequences.

*Theories make a difference in the world not by answering questions, but by guiding intelligent search.*

Of course, a theory cannot "figure out" anything. It is, after all, merely a text. But we can very often build a model on the basis of a theory. And there are models which can, in an entirely nontrivial sense, figure things out. Here I am not referring to static scale models, like those made by architects to show clients what their finished buildings will look like. Nor do I mean even the scale models of wings that aerodynamicists subject to tests in wind tunnels; these are again static. However, the system consisting of both such a wing and the wind tunnel in which it is flown is a model of the kind I have in mind. Its crucial property is that it is itself capable of behaving in a way similar to the behaving system it represents, that is, a real airfoil moving in a real airmass. The behavior of the wing in the wind tunnel is presumably determined by the same aerodynamic laws as govern the behavior of the wings of real airplanes in flight. The aerodynamicist therefore hopes to learn something about a full-scale wing by studying its reduced-scale model.

The connection between a model and a theory is that a model *satisfies* a theory; that is, a model obeys those laws of behavior that a corresponding theory explicitly states or which may be derived from it. We may say, given a theory of a system $B$, that $A$ is a model of $B$ if that theory of $B$ is a theory of $A$ as well. We accept the condition also mentioned by Kaplan that there must be no causal connection between the model and the thing modelled; for if a model is to be used as an explanatory tool, then we must always be sure that any lessons we learn about a modeled entity by studying its model would still be valid if the model were removed.

People do, of course, derive consequences from theories without building explicit models like, say, scaled-down wings in wind tunnels. But that is not to say that they derive such consequences without building models at all. When a psychiatrist applies psychoanalytic theory to data supplied to him by his patient, he is, so to speak, exercising a mental model, perhaps a very intuitive one, of his patient, a model cast in psychoanalytic terms. To state it one way, the analyst finds the study of his mental model ($A$) of his patient ($B$) useful for understanding his patient ($B$). To state it another way, the analyst believes that psychoanalytic theory applies to his patient and therefore constructs a model of him in psychoanalytic terms, a model to which, of course, psychoanalytic theory also applies. He then transforms (translates is perhaps a better word) inferences derived from working with the model into inferences about the patient. (It has to be added, lest there be a misunderstanding, that however much the practicing psychoanalyst is committed to psychoanalytic theory and however much his attitudes are shaped by it, psychoanalytic therapy consists in only small part of direct or formal application of theory. Nevertheless, it is plausible that all of us make all our inferences about reality from mental models whose structures, and to a large extent whose contents as well, are strongly determined by our explicitly and implicitly held theories of the world.)

Computers make possible an entirely new relationship between theories and models. I have already said that theories are texts. Texts are written in a language. Computer languages are languages too, and theories may be written in them. Indeed, for the present purpose we need not restrict our attention to machine languages or even to the kinds of "higher-level" languages we have discussed. We may include all languages, specifically also natural languages, that computers may be able to interpret. The point is precisely that computers do *interpret* texts given to them, in other words, that texts determine computers' behavior. Theories written in the form of computer programs are ordinary theories as seen from one point of view. A physicist may, for example, communicate his theory of the pendulum either as a set of mathematical equations or as a computer program. In either case he will have to identify the terms of his theory—his "variables," in technical jargon—with whatever they are to correspond to in reality. (He may say $l$ is the length of the pendulum's string, $p$ its period of oscillation, $g$ the acceleration due to gravity, and so on.) But the computer program has the advantage not only that it may be understood by anyone suitably trained in its language, just as a mathematical formulation can be readily understood by a physicist, but that it may also be run on a computer. Were it to be run with suitable assignments of values to its terms, the computer would *simulate* an actual pendulum. And inferences could be drawn from that simulation, and could be directly translated into inferences applicable to real pendulums. A theory written in the form of a computer program is thus both a theory and, when placed on a computer and run, a model to which the theory applies. Newell and Simon say about their information-processing theory of human problemsolving, "the theory performs the tasks it explains." Strictly speaking, a theory cannot "perform" anything. But a model can, and therein lies the sense of their statement. We shall, however, have to return to the troublesome question of what the performance of a task can and cannot explain.

In order to aid our intuition about what it means for a computer model to "behave," let us briefly examine an exceedingly simple model: We know from physics, and indeed it follows from the equation $f = ma$ that we mentioned earlier, that the distance $d$ an object will fall in a time $t$ is given by:

$$d = at^2/2$$

where $a$ is the acceleration due to gravity. In most elementary physics texts, $a$ is simply asserted to be the earth's gravitational constant, namely, $32 \text{ ft/sec}^2$, where the unit of distance is feet and that of time is seconds. The equation itself is a simple mathematical model of a falling object. If we assume, for the sake of simplicity, that the acceleration $a$ is indeed constant, namely, $32 \text{ ft/sec}^2$, we can compute how far an object will have fallen after, say, 4 seconds: $4 \times 4 = 16$ and $16 \times 32 = 512$. The answer, as schoolchildren would say, is therefore 512 feet.

*There are models which can, in an entirely nontrivial sense, figure things out.*

Mathematicians long ago fell into the habit of writing the so-called variables that appear in their equations as single letters. Perhaps they did this to guard against writer's cramp or to save chalk. Whatever their reasons, their notation is somewhat less than maximally mnemonic. Because computer programs are often intended to be read and understood by people, as well as to be executed by computers, and since computers are, within limits, indifferent to the lengths of the symbol strings they manipulate, computer programmers often use whole words to denote the variables that appear in their programs. Other considerations make it inconvenient to use juxtaposition of variables, as in $xy$, to indicate multiplication. Instead the symbol "*" is used in many programming languages. Similarly, "**" is used to indicate exponentiation. Thus, where the mathematician writes $t^2$, the programmer writes `t**2`. The equation:

$$d = at^2/2$$

when transformed into a program statement may thus appear as:

```
distance = (acceleration
                 *time**2)/2.
```

A significant technical point must be made here. Although the "statement" shown here is a transliteration of the equation to which it corresponds, it is not itself an equation. In technical parlance, it is an "assignment statement." It assigns a value to the variable "distance." "Distance," in turn, is technically an "identifier," the name of a storage location in which is stored the value which has been assigned to the corresponding variable. In mathematics, a variable is an entity whose value is not known, but which has a definite value nontheless, a value that can be discovered by solving the equation. In programs, a variable may have different values at different stages of the execution of the program. In ordinary mathematics, e.g., in high-school algebra, the "equation" "$x = x + 1$" is nonsense. The same string of symbols appearing as an expression in a program has meaning, namely, that 1 is to be added to the contents of the location denoted by "$x$" and those contents replaced by the resulting sum.

*A model is always a simplification, a kind of idealization.*

Let us now complicate our example just a little. Suppose an object is to be dropped from a stationary platform, say, a helicopter hovering at some altitude above the ground. The object's height above the ground after it has fallen for some time would then be given by:

```
height = altitude
      — (acceleration*time**2)/2
```

Finally, suppose that the helicopter is flying forward at some constant velocity while maintaining its altitude. If there were no aerodynamic effects on the object dropped from the helicopter, it would remain exactly below the helicopter during its entire journey to the ground. The object's horizontal displacement from the point over which it was dropped would therefore be the same as the helicopter's horizontal displacement from that point, that is:

```
displacement = velocity*time
```

where by "velocity" we here, of course, mean the helicopter's velocity.

We now have, from one point of view, two equations, from another point of view, two program statements, from which we can compute the horizontal and vertical coordinates of an object dropped from a moving helicopter. We can combine them and embed them in a small fragment of a computer program, as follows:

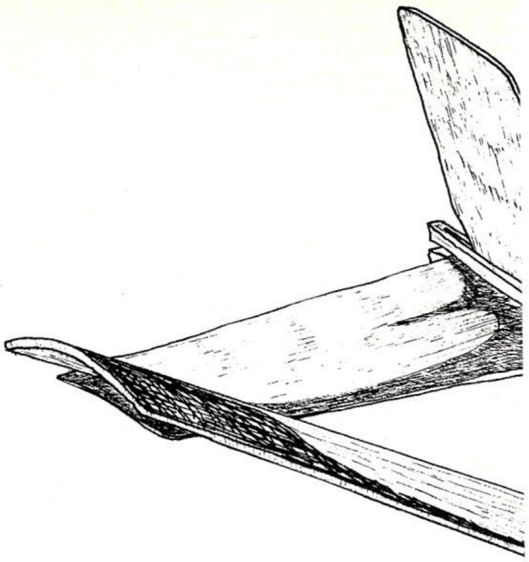
```
FOR time = 0 STEP .001 UNTIL
                    height = 0 DO;
  height = altitude
    — (acceleration*time**2)/2;
  displacement = velocity*time;
  display (height, displacement);
END.
```

This is an example of a so-called *iteration statement*. It tells the computer to do a certain thing until some condition is achieved. In this case, it tells the computer to first set the variable "time" to zero, then to compute the height and displacement of what we would interpret to be the falling object, then to display the coordinates so computed—I shall say more about displaying in a moment—and, if the computed height is not zero, to add .001 to the variable "time" and do the whole thing again, that is, to iterate the process. (This program contains an error which, for the sake of simplicity, I have let stand. As it is, it may run forever. To repair it, the expression "height = 0" should be replaced by "height < 0." The reason for this is left to the reader to discover.)

We have assumed here that the computer on which this program is to run has a built-in display apparatus and the corresponding display instruction. We may imagine the computer's display to be a cathode-ray tube like that of an ordinary television set. The display instruction delivers two numbers to this device, in this example, the values of height and displacement. The display causes a point of light to appear on its screen at the place whose coordinates are determined by these two numbers, i.e., so many inches up and so many inches to the right of some fixed point of origin.

If we now make some additional assumptions about, for example, the persistence of the lighted dot on the screen and the overall timing of the whole affair, we can imagine that the moving dot we see will appear to us like a film of the object falling from the helicopter (see Figure 1). It is thus possible, even compelling, to think of the computer "behaving," and for us to interpret its behavior as modeling that of the falling object.

It would be very easy for us to complicate our example step by step, first, for example, by extending it to cover the trajectory of a missile fired from a gun and, with that as a base, to extend it to the flight of orbiting satellites. We would then have described at least the most fundamental basis on which the orbital simulations we often see on television are developed. But that is not my purpose. Simple as our example is, we can learn pertinent lessons from it.



To actually use the model, an investigator would initialize it by assigning values to the parameters altitude and velocity, run it on an appropriate computer, and observe its behavior on the computer's display device. There would, however, be discrepancies between what the model, so to speak, says a falling object would do and the behavior of its real counterpart. The model, for example, makes the implicit assumption that there are no aerodynamic effects on the falling object. But we know that there would certainly be air resistance in the real situation. Indeed, if the object dropped were a parachute, its passenger's life would depend on air resistance slowing its fall. A model is always a simplification, a kind of idealization of what it is intended to model.

The aim of a model is, of course, precisely not to reproduce reality in all its complexity. It is rather to capture in a vivid, often formal, way what is essential to understanding some aspect of its structure or behavior. The word "essential" as used in the above sentence is enormously significant, not to say problematical. It implies, first of all, purpose. In our example, we seek to understand how the object falls, and not, say, how it reflects sunlight in its descent or how deep a hole it would dig on impact if dropped from such and such a height. Were we interested in the latter, we would have to concern ourselves with the object's weight, its terminal velocity, and so on. We select, for inclusion in our model, those features of reality that we consider to be essential to our purpose. In complex situations like, say, modeling the growth, decay, and possible regeneration of a city, the very

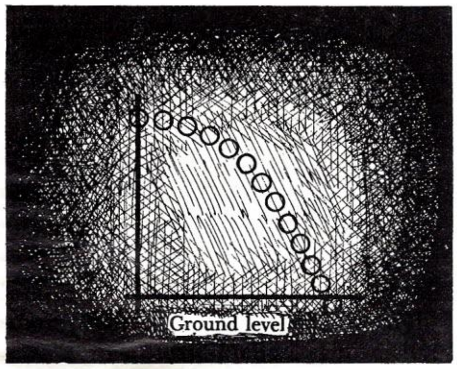
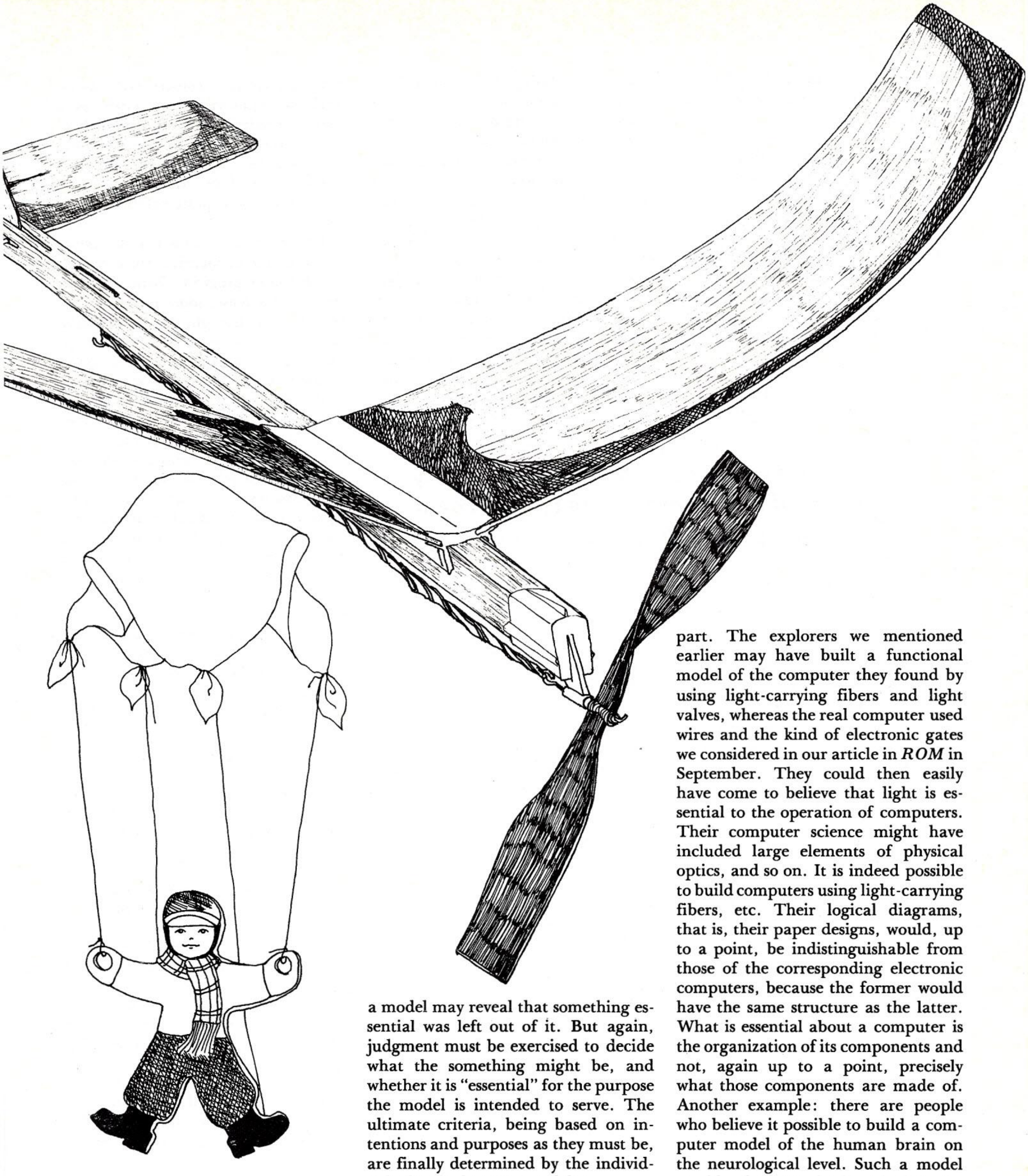


**Figure 1**

**Cathode simulation of the trajectory of an object dropped from a flying helicopter.**



act of choosing what is essential and what is not must be at least in part an act of judgment, often political and cultural judgment. And that act must then necessarily be based on the modeler's intuitive mental model. Testing a model may reveal that something essential was left out of it. But again, judgment must be exercised to decide what the something might be, and whether it is "essential" for the purpose the model is intended to serve. The ultimate criteria, being based on intentions and purposes as they must be, are finally determined by the individual, that is, human, modeler.

The problem associated with the question of what is and what is not "essential" cuts the other way as well. A model is, after all, a different object from what it models. It therefore has properties not shared by its counterpart. The explorers we mentioned earlier may have built a functional model of the computer they found by using light-carrying fibers and light valves, whereas the real computer used wires and the kind of electronic gates we considered in our article in *ROM* in September. They could then easily have come to believe that light is essential to the operation of computers. Their computer science might have included large elements of physical optics, and so on. It is indeed possible to build computers using light-carrying fibers, etc. Their logical diagrams, that is, their paper designs, would, up to a point, be indistinguishable from those of the corresponding electronic computers, because the former would have the same structure as the latter. What is essential about a computer is the organization of its components and not, again up to a point, precisely what those components are made of. Another example: there are people who believe it possible to build a computer model of the human brain on the neurological level. Such a model would, of course, be in principle describable in strictly mathematical terms. This might lead some people to believe that the language our nervous system uses must be the language of our mathematics. Such a belief would be an error of the kind we mean. John von Neumann, the great computer pioneer, touched briefly on this point himself:

> "When we talk mathematics, we may be discussing a *secondary* language, built on the *primary* language truly used by the central nervous system. Thus the outward forms of *our* mathematics are not absolutely relevant from the point of view of evaluating what the mathematical or logical language *truly* used by the central nervous system is."

One function of a model is to test theories at their extreme limits. I have already mentioned that computers can generate films that model the behavior of a particle at extreme limits of relativistic velocities. Our own simple model of falling objects could be used in its present form to simulate, hence to calculate, the fall of an object from a spaceship flying near the surface of the moon. All we would have to do is to initialize acceleration to the number appropriate for the gravity existing on the moon's surface (providing, of course, that the spaceship is not so high above the surface of the moon that the effect of the moon's gravitational field would have been significantly changed—another implicit assumption). For that simulation exercise we would not have to have any components in our model corresponding to air resistance or other aerodynamic effects: the moon has no atmosphere. (Recall that an astronaut simultaneously dropped a feather and a hammer onto the moon's surface and that they both reached the ground at the same time.)

It is a fact, however, that the moon's gravitational field varies from place to place. These variations are thought to be due to so-called masscons, that is, concentrations of mass within the moon that act somewhat like huge magnets irregularly buried deep within the moon. The masscon hypothesis was advanced to account for observed irregularities in the trajectories of spacecraft orbiting the moon. It is, in effect, an elaboration of the falling-body model we have discussed. The elaborated model is the result of substituting a complex mathematical function (in other words, a subroutine) for the single term "acceleration" of our simple model. I mention it to illustrate the process, in this case properly applied, of elaborating a model to account for new and unanticipated observations. But the masscon elaboration was not the only possible extension of either the theory or its computer model. It could have been hypothesized, for example, that the moon is surrounded by a turbulent ether mantle whose waves and eddies caused the spaceship's irregular behavior. There are dozens of very good reasons for rejecting this hypothesis, of course, but a good programmer, given a lot of data, could more or less easily elaborate the model with which we started by adding "ether turbulence subroutines" so that in the end, the model behaved just as the spaceship was observed to behave. Such a model would, of course, no longer look simple. Indeed, its very complexity, plus the precision to which it carried its calculations, might lend it a certain credibility.

*The very eloquence of theories, especially in the eyes of their authors, may give them a persuasive power they hardly deserve.*

Earlier I said that the value of a theory lies not so much in the aggregation of the laws it states as in the structure that interconnects them. The trouble with the kind of model elaboration that would result from such an "ether turbulence" hypothesis is that it simply patches one more "explanation" onto an already existing structure. It is a patch in that it has no roots in anything already present in the structure. Computer models have, as we have seen, some advantages over theories stated in natural language. But the latter have the advantage that patching is hard to conceal. If a theory written in natural language is, in fact, a set of patches and patches on patches, its lack of structure will be evident in its very composition. Although a computer program similarly constructed may reveal its impoverished structure to a trained reader, this kind of fault cannot be so easily seen in the program's performance. A program's performance, therefore, does not alone constitute an adequate validation of it as theory.

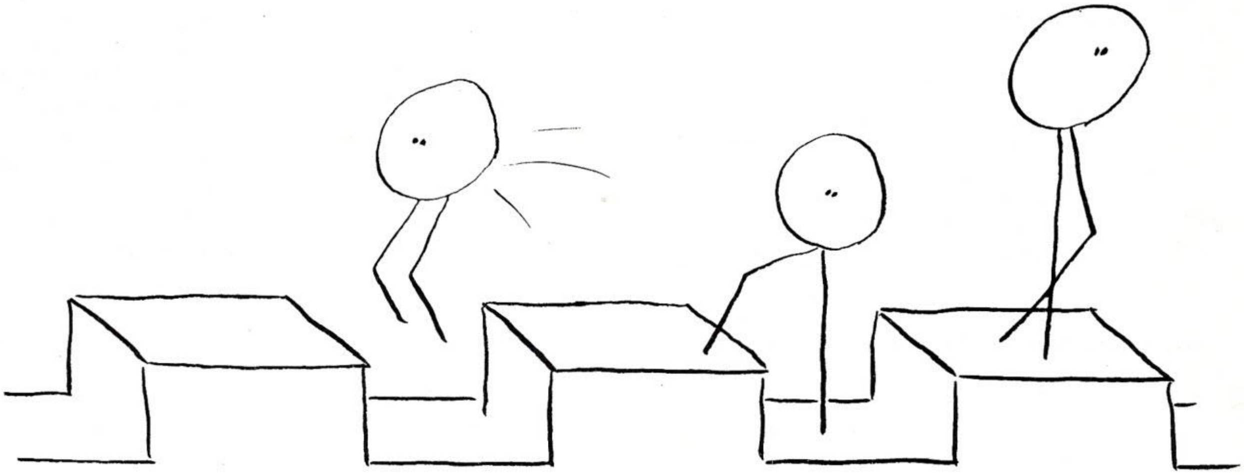
I have already alluded to the heuristic function of theories. Since models in computer-program form are also theories (at least, some programs deserve to be so thought of), what I have said about theories in general also applies to them, perhaps even more strongly, in this sense: in order for us to draw consequences from discursive theories, even to determine their coherence and consistency, they must, as I have said, be modeled anyway, that is, be modeled in the mind. The very eloquence of their statements, especially in the eyes of their authors, may give them a persuasive power they hardly deserve. Besides, much time may elapse between the formulation of a theory and its testing in the minds of men. Computer programs tend to reveal their errors, especially their lack of consistency, quickly and sharply. And, in skilled hands, computer modeling provides a quick feedback that can have a truly therapeutic effect precisely because of its immediacy. Computer modeling is thus somewhat like Polaroid photography: it is hard to maintain the belief that one has taken a great photograph when the counterexample is in one's hands. As Patrick Suppes remarked:

> "The attempt to characterize exactly models of an empirical theory almost inevitably yields a more precise and clearer understanding of the exact character of a theory. The emptiness and shallowness of many classical theories in the social sciences is well brought out by the attempt to formulate in any exact fashion what constitutes a model of the theory. The kind of theory which mainly consists of insightful remarks and heuristic slogans will not be amenable to this treatment. The effort to make it exact will at the same time reveal the weakness of the theory."

The question is, of course, just what kinds of theories are "amenable to this treatment?" ▼

# *What is a* MICROCOMPUTER SYSTEM?

### by Leslie Solomon and Stanley Veit



A few years ago small computers did not exist; today they are moving into the lives of everyone. No one can quite predict the changes they will cause or the form they will take. However, the fact that just about everyone will have to know something about what a computer is and how it works is indisputable.

Five elements, combined, make up a computer system:

1. The central processing unit (CPU), which in the case of a home computer using a chip is commonly called the "microprocessing unit" (MPU) or "microprocessor."
2. The memory, which stores computer programs and the results of calculations.
3. The input/output (I/O) ports through which data enter and leave the computer.
4. The clock, which causes things to happen in sequence.
5. The power supply, which provides the direct current to the computer.

In addition, there has to be some organization of the various wires or other current-carrying leads that connect the parts of the computer. The arrangement of these leads is called a bus.

## Clock

Because a computer system is a complex arrangement of digital hardware, marching to the tune of the software, something has to establish the correct "beat" so that hardware and software can "do their thing" in step. Without correct timing all would be chaos.

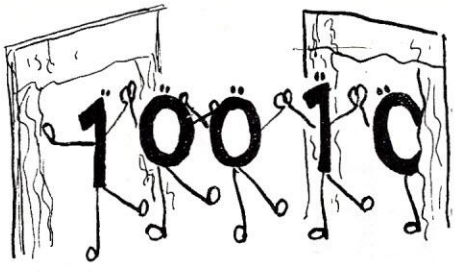
The beat in a computer is generated by an electronic circuit called a clock, which emits an accurately controlled electrical pulse at certain prescribed intervals. Clock timing is usually controlled by a quartz crystal (like a digital wristwatch) which has excellent stability and will sustain the correct beat for extremely long periods.

The clock signal is then passed to all devices on the bus (except the power supply) and everyone has the same time reference.

---

From *Getting Involved With Your Computer,* Copyright © 1977 by Ridley Enslow Publishers.

Illustrated by Korkie



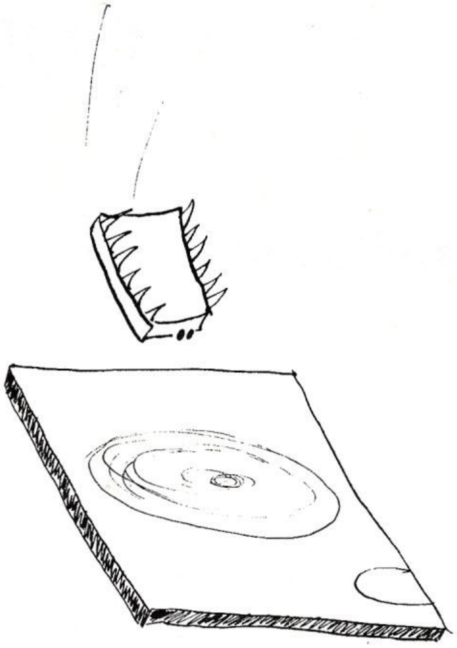
## I/O

The interface between a computer and the outside world is called a port. Because most ports you will see in hobby computers are both input and output, they are called I/O ports.

The input (I) portion accepts the data from the external device—for example, a keyboard or tape reader—and then prepares them for use by the computer. The output (O) portion accepts the processed data from the computer and prepares them for use by whatever device is electrically connected—perhaps a TTY or CRT terminal, for example.

There are two types of I/O port, serial and parallel. A serial port accepts data from the external device in serial fashion (one bit at a time, sort of digital Indian file) and returns the processed data to the associated output device in the same way. Special circuits in the port have already been set up to determine the speed (called baud rate) of the serial transmissions.

Parallel ports accept and deliver all the multibit data words at one time; therefore these ports operate much faster than serial ports. Some computers use a combination in one device; for example, a keyboard will input data in a parallel fashion and a port will output data in serial fashion for use by a serial device such as a TTY or CRT terminal.



## The Microprocessor

A basic microprocessor consists of a number of specialized logic circuits, all arranged on a single semiconductor "chip" called a microprocessor—sometimes called an MPU (microprocessing unit) or CPU (central processing unit). Figure 1 is a simple block diagram.

The thick arrows that interconnect certain blocks mean that several lines join these items.

This device contains a number of registers (storage areas) and a master control system. The ALU (arithmetic logic unit) performs digital addition and subtraction of two input "words" and can detect such things as equality between them.

Two numbers are added by placing them in separate registers, then adding (or subtracting) them and placing the result in another register. If there is any "overflow" from the arithmetic operation, a "carry," called a flag, is generated.

Figure 2 describes the basic approach. Here, we are going to add 1 + 1. We are using a four-bit "word" in this illustration. In the first column (1) we have a 1 and a 0. If they are added, the result is a 1, which is placed in the third register (bottom). The next column (2) contains another 0 and 1 whose total is also 1 and is placed in its box in the bottom register. The other two columns (4,8) contain 0s; their total is 0. The result, shown in the lowest storage register, is 0011, the binary equivalent of 3. (Count up the values in the 1, 2, 4, and 8 columns.)

To illustrate a "carry" Figure 3 shows the addition of 1 + 3. In the 1 column we have two 1s, and because they add up to a sum greater than can be expressed in a single binary bit (remember that binary has only two digits—0 and 1), the total becomes 0 with a 1 to "carry" over. This makes the next column (2) a 1, plus the 1 from the carry. Again this total is

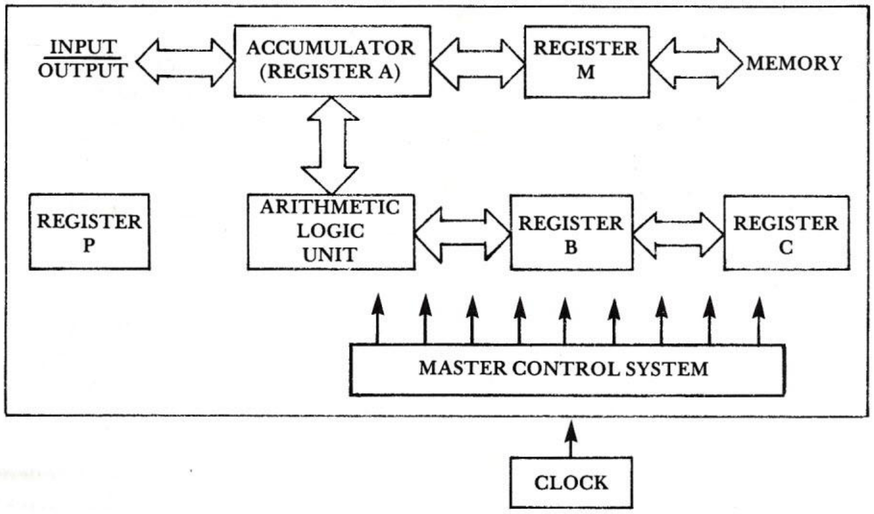


**Figure 1**
**Simplified block diagram of a microprocessor. A double-ended arrow indicates that the signal passes in both directions. A broad arrow indicates multiple lines. The clock synchronizes the operation of the system.**

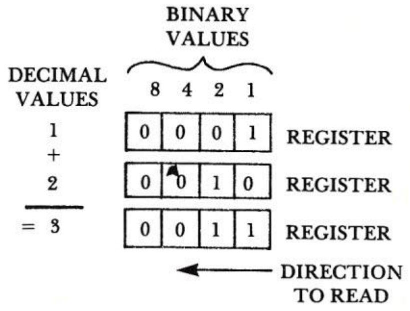


**Figure 2**
**Simple binary addition.**

greater than a binary 1 and we make this column a 0 with a 1 carry over. The third column (4) contains two 0s; therefore a 0 plus a 1 equals a 1. The total of 0001 plus 0011 equals 0100—4 in the binary code.

A processor can contain many registers, referred to as data, working, or scratch-pad memories and used by the processor for temporary data storage (usually for intermediate results during data manipulation). Another register, (register-P) is used as the program counter which keeps track of each step that the processor will follow in the prescribed series of operations.

The accumulator register is one of the most important sections in the processor because, as you can see in Figure 1, all incoming and outgoing data must pass through it. The accumulator can also do other things to the data, too complex to cover in an introductory article.

The control unit is a group of elements that provides "instructions" to the remainder of the logic circuits to maintain proper control. The control unit gets its orders from the computer instructions inserted by the user. You can visualize the control unit as a traffic cop handling traffic at a busy intersection. This operations center of the processor sets up the sequences in machine cycles (pulses); that is, the entire program is controlled by an electronic clock and all elements do their thing to its "beat." In this way order is maintained.

The processor's two basic operations are called fetch and execute. In the fetch operation data stored in some part of the memory are "fetched" and placed in a register. The next step—execute—then performs the operation selected by the pertinent instruction. In Figure 4 you can see how the processor fetches and executes in step with the machine clock (cycles). Sometimes the fetch cycle is extended slightly by what is called a wait state to allow the memory enough time to locate the requested data. Keep in mind that some memory chips are slower than others. The wait state allows slower memories to work properly within the system. The speed of memory operation is called access time and is usually specified in nanoseconds.

Not all processor instructions are 8-bits (eight bits equal one byte) and some are two or three bytes long; for example, you cannot tell a computer to "input" something unless you specify where that something is and what to do with it. The 8080 instruction to input is DB (in hex). However, you must also tell the computer what to input and where it is; therefore more instructions are needed. An example of this is DB 01, which means (in 8080 language) accept an input from input port 1. By the same token you cannot tell the processor to output something unless you specify what you want outputted and where. If you want the computer to "jump" to a particular memory location, you must tell it to jump (C3 in 8080 lingo) and then insert the address—C3 00 10, for example, means to jump to memory address 1000 (hex). Note that the jumped-to address is backward, with the least significant digits (00) placed first and the most significant digits (10) last. This "backward" approach is a feature of several processors.

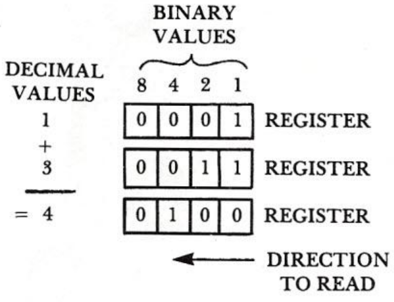


**Figure 3**
**Binary addition illustrating carry.**

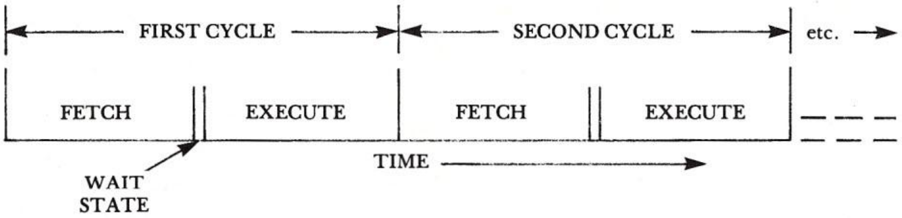
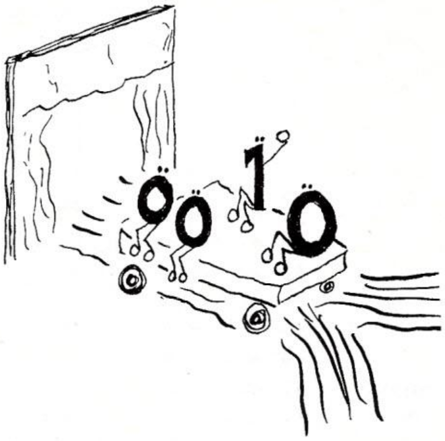


**Figure 4**
**The fetch/execute cycle.**



## The Bus Structure

A bus is a group of leads that carries all the electrical signals to and from the processor. The leads may be wires, copper traces, or a printed circuit board. The Altair, IMSAI, Sol, and some other computers use the S-100 which, because it was built by MITS, the developer of the original Altair, is commonly called the Altair bus.

The S-100 is formed by a 100-line set of copper foil traces on a large printed circuit (PC) board, sometimes called a mother board. These lines (not all of which are in use at once, some being available for future expansion) are electrically connected to the CPU board, clock, and power supply. The user can then "plug" twenty or so devices such as memories and I/O ports into the bus merely by inserting the selected device into one of the 100-pin connectors. The actual signals carried on the Altair and similar bus structures are shown in Figure 5. Keep in mind that the hobby computer demand is growing rapidly and that this bus structure may be modified by the time you have read this description.

The "pin out" of the 8080 is shown in Figure 5 and the reader can see how the 8080 mates with the classical computer bus structure in Figure 6.

The bidirectional DATA bus carries the D0 through D7 inputs and outputs from the CPU to the outside world or from the outside world to the CPU. Special chips in the circuit allow the data to travel both ways on this bus without interference. This is accomplished essentially by a form of high-speed electronic switching and precise timing (from the clock); the electronic switches are set up to allow the data

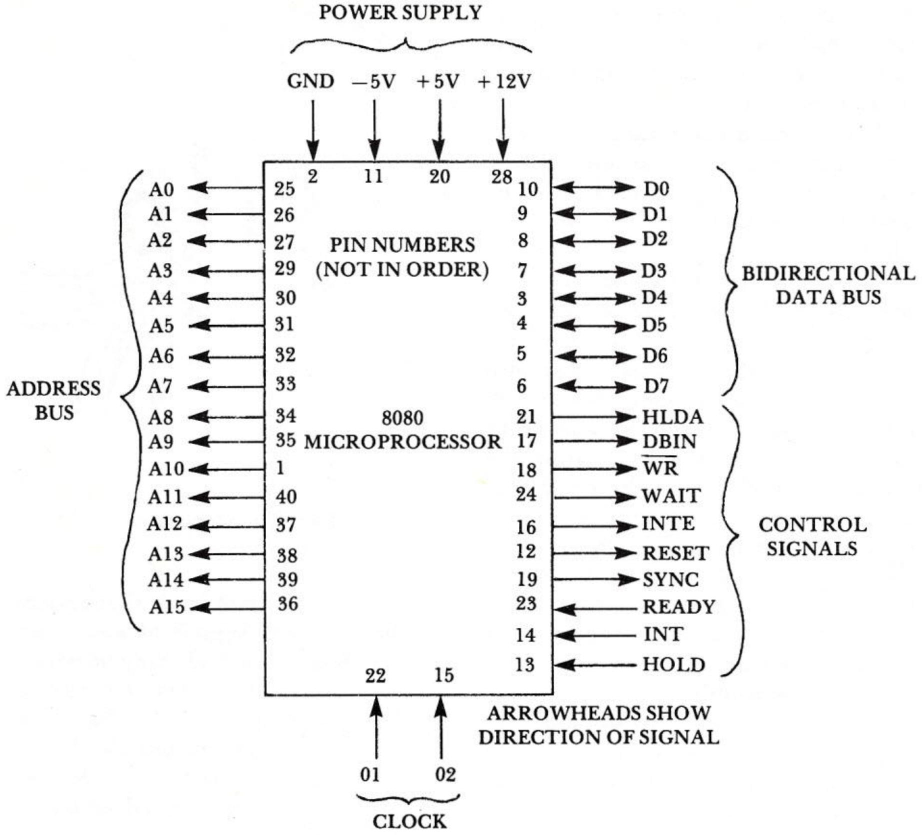

POWER SUPPLY
GND
−5V
+5V
+12V
2
11
20
28
ADDRESS BUS
A0 25
A1 26
A2 27
A3 29
A4 30
A5 31
A6 32
A7 33
A8 34
A9 35
A10 1
A11 40
A12 37
A13 38
A14 39
A15 36
PIN NUMBERS (NOT IN ORDER)
8080 MICROPROCESSOR
10 D0
9 D1
8 D2
7 D3
3 D4
4 D5
5 D6
6 D7
BIDIRECTIONAL DATA BUS
21 HLDA
17 DBIN
18 $\overline{WR}$
24 WAIT
16 INTE
12 RESET
19 SYNC
23 READY
14 INT
13 HOLD
CONTROL SIGNALS
22
15
ARROWHEADS SHOW DIRECTION OF SIGNAL
01
02
CLOCK

**Figure 5**
**8080 pin identification ("pin out"). The 8080 microprocessor has 40 pins, each carrying its own signal in or out of the chip. This chart identifies each pin by number and signal.**

lines to "output" from the CPU at a certain time before switching over to permit the CPU to receive data.

The ADDRESS bus is only one way—from the CPU to the external circuits. Everything tied to the ADDRESS bus has its own unique "address," a special bus-coupling circuit wired so that only that specific address will allow data to pass either way—no other address will work.

This arrangement permits the CPU to select the device or memory with which to "talk" or "listen" and reject all others. Most hobby computers can handle a couple of hundred external devices.

*The entire program is controlled by an electronic clock; all elements do their thing to its "beat."*

The CONTROL bus contains the "handshake" signals that tell the non-CPU devices what to do at a particular instant. As shown in Figure 6, the control signals include WR (write-read), which tells the memory in which direction the data must flow, RESET to clear everything in preparation for data manipulation, and WAIT to inform the system that for some reason processing has been halted.

The CLOCK line is used to supply the system with the timing signals to give everything on the bus the same time reference; thus all operations will occur in the proper order. The POWER SUPPLY bus provides the required DC power needed by the devices to operate.

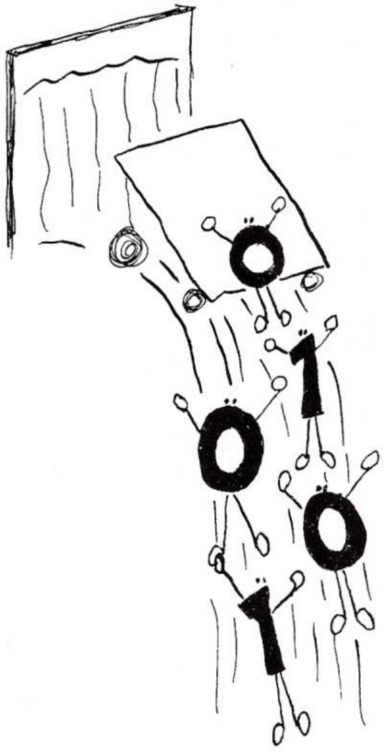
Literally dozens of microprocessors are available to the hobbyist. Each type has its own adherents who swear by it (and sometimes at it) and each "does its own thing."

At the time of this writing, and most likely for the next several years, the two most popular types of processor are the 8080 (including the Z80) and the 6800—both available from a number of semiconductor manufacturers. Each is well supported by peripheral "chips" and software.

An in-depth discussion of the "innards" of a microprocessor is beyond the scope of this basic treatment, but for all intents and purposes they operate in the same way, differing only in "instruction set" and power requirements.

Other computers use buses of different forms, both electronically and mechanically. Therefore you cannot, for example, use Altair-compatible plug-ins with a SWTPC (Southwest Technical Products) 6800 computer and vice versa. Several companies are currently making a wide range of plug-ins for the SWTP-6800 bus structure.

Now you can see that if you are planning to expand your system at some future date it will be best to look for a computer with a supported bus structure in order to take advantage of the many peripherals that can be used with it.



## How a Microcomputer Works

The microprocessor shown in Figure 6 has five basic elements: The MPU (microprocessor unit) which does all the data processing, some RAM memory (random access memory also called "read-write memory"), an I/O

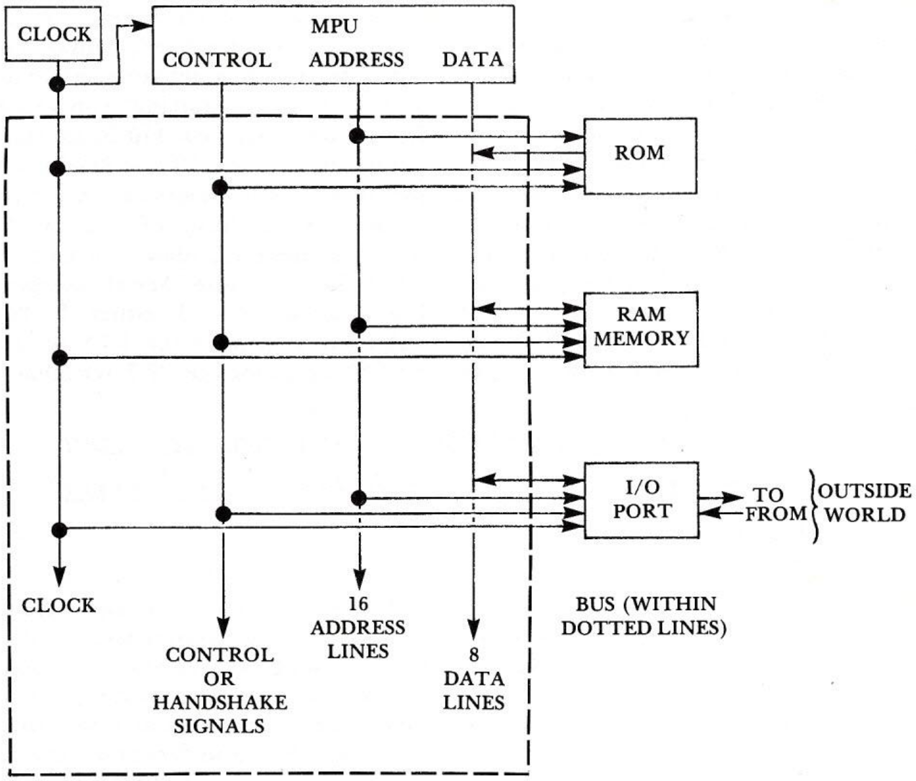


**Figure 6**

**A microprocessor system: five basic elements plus the bus structure. The bus structure (everything within the dotted lines) carries all interconnecting address, data, "handshake" control, power, and ground signals.**

port (input-output port for data), and the clock. In the basic system shown, an "operating system" is residing in ROM memory. The operating system is a program that directs the entire system in it's operation. It is sometimes called an "executive" program. The ROM memory has the capability of storing the program even after the power is turned off. However, like the printed page of this book, it can only be read from, nothing can be changed, or added.

The general operation of the CPU and clock were discussed in the preceding paragraphs and are not covered here.

The bootstrap program in the ROM makes sure that when the computer is first turned on the CPU will automatically "look" at the ROM data for its instructions.

One set of instructions may tell the computer which I/O port to look at first and what to do with the data entering through that port.

In I/O instruction the computer sets up the correct address for that I/O port and then via a set of "handshake" signals turns on that port to make it ready to accept data. The incoming data are then deposited on the data lines. Simultaneously the instructions have told the processor to open the "write" lines to the particular RAM memory. These instructions have also provided the address of that memory, and when this portion of the memory receives the correct address in conjunction with the "handshake" signals the incoming data are deposited in it.

At the conclusion of the data input the processor shuts down the I/O port until it is needed again. The input data are now stored in memory and await use by the operator.

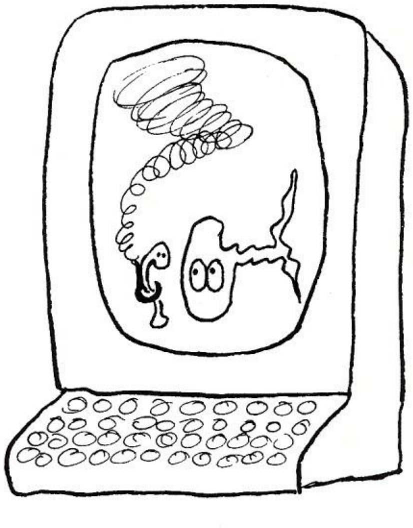
Other instructions stored in the ROM may allow the user to "dump" the stored data into a selected I/O port for rechecking against the original. This "dump" does not destroy the stored data, for only the "read" line was activated. It allows the data to be displayed only on the selected output device.

To perform the dump the processor simultaneously transmits the address down the address lines, where only the selected device can "pick off" these data, and sends a "read" signal down the control lines. When the two signals coincide, only that particular memory will dump its data onto the data bus. At the same time another address is sent down the line with the appropriate control signals to "open up" the selected I/O port and arrange it so that it outputs data instead of receiving them. The data are now sent to the output device (printer or CRT terminal).

Operating systems may also include many other "commands," necessary to the operation of the computer system.

Thus you can see that under the control of the operating system any element in the computer can be "called" and made to perform its function. The clock makes sure that everything is done "in step" for a smoothly flowing system operation.

In essence, what we are saying is that all elements in the computer that are tied to the address and data lines can be "turned on" by the processor and data can be manipulated between any two elements (or more in some cases) connected to the common bus system. All the processor has to do (under instruction) is send the control signals to the selected device and the data will flow in either direction.



## System Modules

### Audio Cassette Data Storage

An ordinary Phillips-type tape cassette is used with a conventional audio cassette recorder to store data for future use in the computer. To store data it is converted into audio tones which are then recorded in the tape cassette. There is no standard system of recording the data, however, and data recorded by one system cannot be used by another. The methods with the greatest amount of software support are the following:

1. "Kansas City" or Byte/Lancaster. This system is used by other available interfaces (an interface is an electronic circuit that enables the computer to communicate with external devices) and is accurate but somewhat slow.
2. Tarbell. A popular fast interface that uses a "biphase" method of recording. It loads data fast, has lots of software support, and requires selection of tape recorder units.
3. "ACR" Altair. A method used by MITS on the Altair. Not used by other manufacturers. It is supported only by MITS.
4. Morrow. A popular cassette interface that has a program ROM to simplify its use.
5. CUTS (Computer Users Tape System). A new system developed by Processor Technology for use in the Sol. It has reliable medium speed and good software support. It is available for all Altair (S-100) bus machines in a program called CUTTER.
6. The Digital Group. A system (or several systems) developed by Dr. Suding of The Digital Group. Available for both The Digital Group and other computer systems.
7. Other. The above are only a few of the audio-recording mass storage systems in use. The point in selecting a method is to settle on one that is reliable and has good software support. Once you have made your selection dedicate at least one cassette recorder unit to your computer for this task.

### Video Interface Boards

Video interface boards (e.g., the VDM-1) are essentially CRT (cathode ray tube) terminals without the CRT and sweep systems. These boards plug directly into the Altair bus, deliver monochrome (black and white) alphanumeric characters as "video," which can be applied to a conventional CRT monitor or, via an FCC-approved RF modulator, directly to the antenna inputs of a conventional black and white TV receiver. The VDM-1 is controlled by the computer and forms the O of the I/O port; a keyboard or TTY can be used for the I. This is a low-cost way of producing alphanumerics. If color is your thing, then the Dazzler Video Interface Board, which also plugs directly into the Altair bus, can produce a number of dazzling chromatic displays by using a conventional color TV receiver as the video monitor. All video interface board instruction manuals explain how to connect to a TV set or monitor.

*Each type of microprocessor has its own adherents who swear by it (and sometimes at it).*

Some of the video interface boards also have a parallel I/O port to permit direct connection of a keyboard. The Polymorphic Video Interface is one, which in a video I/O system may be the only I/O board required. The Digital Group includes the cassette and video interfaces on one board. The MERLIN is a complete video system that produces both alphanumeric output and video graphics. In addition, MERLIN has its own ROMs to which a system software monitor may be added.

### I/O Boards

Three types of I/O interface board are currently available: parallel, serial, and analog. Many boards are multipurpose; that is, they have a combination of serial and parallel ports on one board (e.g., the 3P + S by Processor Tech and the MIO by IMSAI).

Parallel boards allow for high-speed data transfer; this approach accesses all the data lines at one time and can dump data at high speeds. Serial boards, on the other hand, convert the computer data lines into one serial line and the speed of "talking" can range from a relatively slow 110 baud (bits per second) for the TTY to 9600 baud for some CRT terminals and tape readers. Some form of "on-board" wiring connections allow the user to select the baud rate. Serial interface boards usually provide either the 20-mA current loop for the TTY or the RS-232 signal for the CRT terminal.

Analog interface usually means analog-to-digital conversion. These systems change analog (slow-moving) signals such as potentiometers and thermometers into digital words that the computer can understand. Several computer games use joystick potentiometers to move objects around a CRT screen. These joysticks are coupled to the computer by an analog interface. In the not-too-distant future vocal interfaces will be available that will accept analog vocal commands and convert each spoken word into a digital word that the computer can work with.

Digital-to-analog interfaces convert program output into analog signals to provide servo control and relay closures that can be used to turn electric power devices on and off under program control.

### Prototyping Boards

A great number of boards with "universal" foil patterns are available to the experimenter. These boards have both wire-wrap and solder-tail IC sockets that permit the design and construction of circuits that plug directly into a computer bus.

*In the not-too-distant future vocal interfaces will be available to convert each spoken word into a digital word.*

### Extender Boards

To troubleshoot a board in an operating computer without causing a short the circuits must be connected to the meter or scope probe. Extender boards plug into the bus and raise the board under test above the level of all others. One Altair bus extender, made by Blastmasters (also called Mullen), has a logic probe right on the extender board.

### Floppy Disks

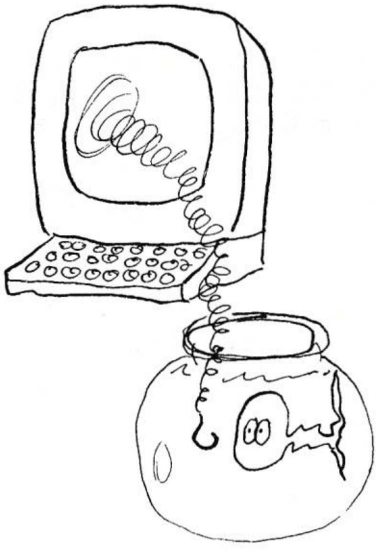
As the personal computer develops into the "small business" computer a better method of mass data storage becomes necessary. The floppy disk fills the requirement for a large-volume, fast-access, easy-to-use system. It comes in "standard" and "mini-floppy" versions and is now on the market at prices the hobbyist can afford. Minifloppy disks are now on the market that use double-density recording and even quad-density recording. This technique extends the data storage capacity of the small disks beyond the capacity of the larger version.

### Other Boards

Solid-state TV-like cameras have also made their appearance. The Cyclops by Cromemco can easily be interfaced with a digital computer and used for character recognition or in a security monitoring system. More than likely some enterprising experimenters will come up with a TV camera/computer interface that will allow the hobbyist to reproduce all the video effects (in color) that are now monopolized by the TV networks.

It is even possible to "update" your computer to one of the newer high-speed and more powerful CPU's (e.g., the Z80). Several manufacturers are now selling a Z80 board that plugs directly into the bus. In the near future multiprocessor boards will permit selection of almost any microprocessor. OSI, for example, is now offering such a board.

Speaking of updating your computer, until now all hobby computers have used eight-bit processors (and will for some years to come). This means eight data lines. In a few years sixteen-bit processors will be with us and we shall then have greater (or more accurate) calculating power. Obviously this means that the present bus structure will no longer be used, for it has room for only eight data lines. Some modification of the bus leads will have to be made or a completely new bus structure designed. However, a set of MPU boards which converts the Altair computer bus into a sixteen-bit machine is made by Alpha Micro Systems Inc. These boards plug directly into the bus and with their software support make the hobby computer into a machine to rival the minis.



### Personal Computing Now and in the Future

The original hobby computer, like many commercial units, looked like a computer because it consisted of a front panel loaded with switches and LED readouts. Examples include the Altair 8800 and the IMSAI 8080. They are still good systems and can still be found in local computer stores.

These computers used their front-panel switches to insert the "bootstrap" (a machine language set of instructions that told the computer which I/O port to look at and what to do with the data coming from that port). This also meant that in addition to the basic computer you needed a plug-in I/O port to suit the terminal. Many other functions could be inserted into the computer by using the front-panel switches with the selected address and the data at that address readily observable on the front panel LEDs. A considerable number of computer hobbyists still feel that front panel switches are "the only way to go."

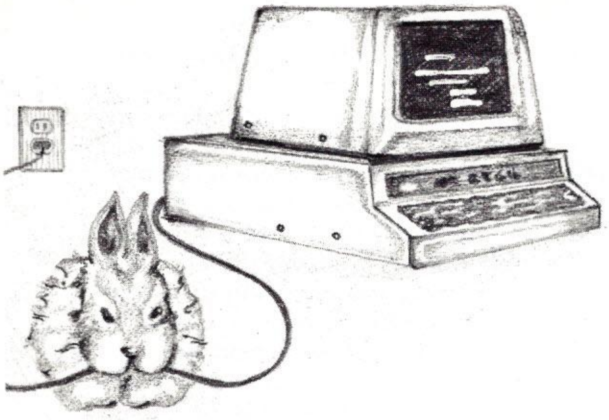
Another development is the computer that has no front panel loaded with switches and LEDs. The SWTPC 6800 is a typical example.

In this approach all the necessary computer instructions are contained in a manufacturer's preprogrammed ROM. All you need to do is to turn the computer power on and a prompter symbol appears on the readout. This prompter can be almost any nonalphanumeric character such as a colon or period. Once the prompter turns up, you type in whatever alphabetical characters the ROM needs for it to go into its monitor routine, such as modifying memory, tape input/output, execution of program, or register examination. Each of these ROM systems has its own built-in program, and all information is supplied with the computer. Thus you can see that a ROM system can be handy if your forte is not switch flipping. Besides, the programs built into the ROM are accurate, and there is no way that the user can blow the instructions or put in the wrong one as when handling front-panel switches.

Because the ROM approach is so convenient, many peripheral manufacturers are presently selling ROM PC boards, complete ROM operating systems built up on Altair-type boards so that hobbyists who have Altair or similar computers can go the ROM route.

The Altair-680b takes advantage of both worlds by incorporating front panel switches with a ROM system.

Recently a new form of computer came on the market. A number of years ago Sphere pioneered the idea of putting an entire computer system into one package. Processor Technology then developed the Sol (named after one of the authors of this article). The Sol has a keyboard, memory slots, video interface, power supply, cassette interface, and a monitor ROM system in one unit. All that needs to be added is a CRT monitor or an FCC-approved RF modulator if you want to use a conventional TV receiver as the video monitor. The Sol package, looking more like an oversized typewriter than a computer, is aesthetically pleasing and would look well in any home or office. Other manufacturers have followed this example including Apple and OSI. Some forthcoming computers such as APF PECOS, Commodore PET, and Radio Shack go one step further and include the CRT. The old image of lights and a box of switches is on its way out. ▼



# Maintaining Your Micro

## by O. S. (The Old Soldier)

So you've taken the plunge and bought your personal computer. You've even solved the software problem in one way or the other. At least you've solved part of it, since there's never enough software. So what's left but fun? Well, there's the little matter of maintenance for instance and oh, yes, troubleshooting.

Most computer equipment is accompanied by at least an operating and maintenance manual and a parts list. These should be kept safe from all harm, and especially safe from disappearance, since quite often they become irreplaceable as soon as the manufacturer changes his product. All other documentation, including purchase orders (what you thought you were buying), invoices and shipping tickets (what you actually got) and all of the tags, bulletins, inspector's slips, etc., form vital entries in the history of your unit and should go with the family jewels.

Maintenance instructions can be either owner-oriented or seviceman-oriented. If they start out "All you need is a long persistence dual trace scope and a 0-60 gigahertz signal generator," read and save it anyway. If they state "No owner service required," be suspicious. If schematics or shop manuals are available, try to get them while your money is fresh in the supplier's pocket. Later they might not remember you.

If you were fortunate enough to have studied the manuals while the installer was around, you have already found out what is meant by "answer-back drum" or "tape runout switch." If not, write down a list of such questions and either nail your serviceman with them the next time he is around or call and ask for an explanation. Remember, you did buy the beast.

Don't expect that these guides will take you step by step through a major overhaul, since even the best maintenance manuals cannot substitute for the years of experience a truly competent serviceperson has acquired, but don't ignore them, either. With the insight you should possess after finishing this article, combined with what you have picked up from the manuals, you can make a positive contribution toward the future happiness of your equipment.

### Listen When It Speaks to You

Most equipment has a characteristic noise when it operates. The true mechanic will carry a mental image of this noise, and quite often will perceive a problem first when the noise changes. An increase in the noise of impact made by a printing mechanism can often signal the loosening of a part that should stay tight. If a previously sharp single noise (*klunk*) should become drawn out (*ka-lunk* or *ka-lunk-lunk*), shut down and look for something that wiggles when it shouldn't and tighten it. If fastening screws are actually missing, as can happen in printing mechanisms subject to high acceleration forces, try, oh try to fish them out of the works before they foul something else.

A variation in the speed of mechanical components (wow and flutter to you hi-fi nuts) can be the first indication of failing bearings, lack of lubrication, or an accumulation of dirt in the wrong place. A fluttering noise, like a card in your bicycle spokes (do kids still do that?), or a scraping or rubbing noise indicates that something which moves is being interfered with. This will usually result from something slipping out of its guides, or a guide or moving component dislocated because of loose mounting screws, or worst of all, a bearing worn sloppy.

If you can observe the mechanisms working, look for something moving in step with the sound, and for anything interfering with the movement. If you must shut the machine down to look, examine for burnished or frayed spots indicative of moving contact. If possible, move the components through their operating cycles by hand and feel for binding at certain points in the cycle. The cure for this type of problem could be simple realigning and tightening, but if your first attempts are not too successful, call for help, since some running clearances must be set with gauges, and you don't want to make the problem worse.

If the problem is a jam of tape or paper, remember that some jams clear out better if the offended medium is

Illustrated by Linda Smythe

backed out rather than run further into the mechanism. After you think you have it all clear, try to piece together the fragments to make sure that you haven't left some residue inside to create another jam.

A squeak is almost universally recognized as a plea for lubrication. To the limit of your ability, respond to it.

### The Nose Knows, Touch Tells

Most people have at least smelled a new car, even if they have never owned one. That characteristic odor is a combination of all the solvents of the materials and finishes of the car, with the slightly unpleasant overtone of paint, oil and undercoating burning off hot surfaces.

Data processing equipment will have somewhat the same initial odor, but after the first week or so, the predominant smell should be that of warm oil, primarily associated with moving parts. If a lubricated point overheats, the warm oil smell will take on an acridity reminiscent of a cheap hamburger joint. Shut down and find the trouble spot before you get to the smoke stage, since this can start bearings melting and fusing.

The smell of burning insulation is, to those who have experienced it, unique and unforgettable. If you haven't experienced it before, bridge the terminals of a six-volt lantern battery with an insulated piece of #28 wire (held with a pair of pliers to avoid burning yourself), and train your nose on the resulting smoke. When that odor comes from electrical or electronic gear, shut it off and start looking for the scorched component. Until you become quite proficient in servicing the equipment, the fix for this will probably require professional assistance.

*Bridge the terminals of a 6-volt battery with an insulated piece of #28 wire and train your nose on the resulting smoke.*

While most electrical equipment may feel unpleasantly warm to the touch in normal operation, very seldom will it be so hot that you can't touch it even for a moment. Learn the normal temperatures of your beast and become suspicious if one changes drastically. This could be a ventilation failure, or clogged filters leading to a ventilation failure, or a change in the value of an electrical component prior to catastrophic failure. Again, if the cause is not readily correctable, call for help.

Vibration is just another aspect of the sound picture, and its investigation and cure should proceed along the same lines. Get help if you need it.

One caution about touch troubleshooting: if you ever experience electrical shock from the exterior of your equipment, shut down and get help quickly, since a potential life-threatening hazard may be developing.

### Unclad Examination

You should know just how far you can go in undressing your equipment before you have hazarded yourself or your equipment. It is often best to have your salesman or serviceman show you how to open access panels and remove shrouds to get at the innards. A few words of caution:

1. Unplug the critter. Energized bare terminals are often exposed when panels or other covers are removed. Do not depend on power switches to protect you. If the unit you are undressing is connected to other equipment, that equipment should also be unplugged or the interconnection removed to avoid sneaky shocks.
2. Remove covers carefully. Often there will be switches or indicator lights fastened to the panel rather than the chassis (sloppy). Possibly there will be interference with components that require removal in a particular sequence. Best if you have watched a pro do it first.
3. Put removed covers and panels in a safe place. Quite often a cover will depend on its attachment to the chassis for rigidity, and it will be very susceptible to injury when unattached. You may be unpleasantly surprised at the cost of a replacement.
4. Save all your fasteners. Much equipment, after several disrobings, is held together with far fewer fasteners than the manufacturer thought necessary. This is, at best, sloppy and could lead to problems with operating clearances, etc.

After you have disrobed your unit as far as you dare, look for scorched components, loose wires, loose parts lying around on the bottom, and any other visual indication that all is not as it was when you last viewed the innards. You will no doubt see much dust, the remedy for which will be discussed later. Look also for foreign items that could have slipped in through cracks: bird, rat, and hornet nests, and bugs and spiders. All of these have been found in operating equipment. All should be removed for the equipment's continuing health.

Do not energize unclad equipment until you have gained much experience with what will happen and what you can then touch. Avoid the temptation to reattach loose wires unless you are positive where they go. Since electronic equipment is made from standard components, it is not uncommon to have a great many unused terminals and sockets just aching to receive your vagrant wire, with possible disastrous results.

Soldering of electronic circuitry inside the equipment is fraught with peril to the unwary, and again best left to the pro, especially if you are under warranty or maintenance contract, since the consequences of a botched solder job are readily apparent. Put it back together right.

### Blessed Be the Ties That Bind

Cords and plugs are the umbilici through which the energy and information flow between the components of your system. Frequent examination of these components can avoid your getting "dumped" at an inopportune moment. Watch for chafing on sharp edges (protect with tape and padding), pinching between or under equipment (relocate), and strains on the terminations caused by moving a component without regard to its attachment to other components (be careful).

If you will pardon a personal note, the author is probably the only one on earth who was knocked off line from his computer by a rabbit. A pet bunny, well on her way to becoming a mother, wandered into my study and took a liking to the taste of the wiring between my terminal and my modem. She had severed several wires in her desire to feather her nest with insulation before I caught her. So far my cat chews only lamp cords, but some day he might develop a taste for computer cables, so I examine them carefully every time I go on line.

Power cords chewed by whatever beast can possess the potential for fire, so watch out.

### Keep It Clean

Cleanliness may be next to Godliness, but in a data-processing environment it is frequently next to impossible. But we still have to try.

While some attempt is usually made to filter the air supplied to data-processing equipment, even if it were one hundred percent effective (and it isn't) there is enough internally generated soil to make cleanup a frequently required chore. Dirt is objectionable because:

- It obstructs the passage of cooling air, allowing component temperatures to rise to the destruct point.
- It collects at bearing points, mixing with the oil to make a very effective abrasive, hastening wearout.
- It accumulates on and around recording and reading heads, changing critical spacing.
- It clogs tape and paper guides, causing uneven feeding and jams.
- It gets your hands messy.

Soil comes from, among other sources:

- Exfoliation of skin and loss of hair.
- Carrying in on street clothing and shoes.
- Filthy habits like smoking.
- Fibers from paper and cardstock.
- Oxide coating rubbed off magnetic tape.

Since soil cannot be kept away, it must on occasion be removed. Ideally, the means of removal should take it entirely away from the equipment, not just knock it into the air to be redeposited later. A vacuum cleaner serves this purpose well when it can reach the dirt. A brush, in conjunction with the vacuum, can clean out hard-to-get-at spots.

Great caution should be exercised if you use compressed air. Air should be dry and oil-free (most compressed air is not) and pressure should be kept very low, so that dust is not forced into bearings and delicate components are not sandblasted or physically blown away. Lint-free wipers can remove surplus oil drips and the dust which is invariably attracted by that oil.

Troublesome soil may sometimes require the use of solvents. Do not use flammable solvents, do not use carbon tetrachloride, and do not use anything not tested for reaction with materials in the equipment. Best results are often obtained with damp soapy sponging followed by damp clear-water sponging, then careful drying. Follow manufacturer's instructions explicitly when cleaning tape heads, since a goof here can dump you properly.

Cleanable filters for forced-air cooling are best cleaned in hot soapy water, rinsed in hot clear water, then sprayed with the sticky stuff available from your local heating contractor for restoring filter dust-catching ability.

Finally, if you find that magnetic tape oxide or paper fibers are accumulating very rapidly, check to make certain that the tape or paper guides are smooth and functioning properly, and consider using an alternate brand of material that might reduce shedding.

### Will a Little Lubricant Help?

Lubrication, not unlike sex, is almost invariably either underdone or overdone. (There, now, after all the "come-on" titles we finally got around to sex. Now that sex is out of the way, back to lubrication.)

Whenever touching parts move with respect to each other, there is a resistance to that movement called friction. Some components, like clutches and drive belts, could not function without friction. In fact, when they slip we are in trouble. Other rotating and sliding components must have as little resistance as possible in order to perform their tasks without excess energy consumption, heat generation, or wear. The best guide about whether to lubricate or to abstain from lubricating any particular part is the maintenance manual. Next best is to observe the pro when he lubes it up.

Lacking either of the above guides, the following rules will usually help keep you out of trouble:

- If one part has to drag another part along with it by friction, like the roller in a friction-fed typewriter or a belt drive, keep the oil away.
- If the touching surfaces are fairly exposed, the motion is comparatively slow, and the mating pressure high, as with a cam, consider a fairly heavy grease with the ability to stay put.
- Gears in gear boxes usually require a lighter grease, of about the consistency of Vaseline.
- Ball and roller bearings use both grease and light oil. The provisions made for lubrication will usually tell you which.
- Surfaces over which paper or magnetic tape slides should never be lubed.

***Don't worry about bunnies in your data-processing environment; other ghastly things can and do stalk you.***

When to lube? Usually every thirty to sixty days, to coincide with your cleaning. How much lube? When it starts to drip, stop oiling and start cleaning up. What kind of lube? Sewing machine oil for the oil points, power tool gearbox grease for the gears, and lithium-based greases for the cams. Avoid graphite, since it is messy and electrically conductive.

### Disasters

The author has already mentioned the affair with the rabbit. Perhaps you don't worry about bunnies in your data-processing environment, but other ghastly things can and do stalk you. Such as:

1. Equipment knocked over, dropped, or banged into.
2. Coffee, Coke, or milkshake spilled into a machine.
3. Roof leaks or sprinkler head opens, soaking the machine.
4. Fire.

Falls and bangs are most likely to happen to semiportable units such as terminals, cassette recorders, and video monitors. If the unit busts all to flinders, you have no problem. If, as is more likely, you dent it, bust the case, or just jar it badly, closer examination should reveal whether you can continue to operate or even turn it on for a test.

These eventualities call for undressing the machine, of course. Check for obvious and visible breakage. Then check for plugs and plug-in circuit boards jarred loose, components bent toward one another so that they touch and short, and bent or displaced mechanical parts. If all looks well, try moving mechanical parts through their cycles by hand to check for rubbing or interference. If all looks well and you can't wait to have it checked out by a serviceman, try putting the unit into service in such a manner that you can shut down at the first suspicion of danger. Observe the performance cautiously until you are convinced that normal operation occurs.

Foodstuffs spilled into a machine can be quite a mess. If you are fortunate enough to be there when it happens, first shut down as quickly as possible, in case blowing fuses have not already done that for you. Second, blot or wipe up as much of the mess as you can reach.

Third, undress the machine and check to see how deeply into the vitals the goop penetrated. If you are lucky, only the mechanical parts were caught. Very carefully wipe and wash away every trace of the substance, wipe dry, spray with WD-40 penetrating oil (protect printed circuit boards and electronic wiring from this), lubricate lube points, and have the serviceman check out the electronics.

*Undress the machine and check to see how deeply into the vitals the goop penetrated.*

If electronic components were soaked, wash the goop off with distilled water, treat mechanical parts as above, and get a serviceman to give it a controlled drying and checkout.

For the machine soaked by the roof leak or sprinkler head, cut it loose from all power, WD-40 it, and go for the professional dryout and checkout. Amateurs should never themselves try to dry out a system, since too much heat (even direct sun) can ruin components, and hidden water not eliminated can short or rust out the works.

In case of fire, if the fire is inside the equipment, call your insurance man and forget it. Almost any occurrence beyond minor smoking will generate problems far beyond the first aid described here.

If the fire is outside the equipment, a quick undressing can tell whether the heat got to the works. Unless insurance is involved here, you might try a cleanup and see if it gets you going. Even if it doesn't, the repairman will appreciate working on a clean machine.

### Dear Diary

Earlier, we discussed the importance of keeping all the documentation that came with your machine. Another set of documents just as critical is a running diary or log of everything that happens to the machine. While running time is nice to know, more important are cleaning and repair records (just how often did that capstan drive fail?), records of all modifications including corrected schematics (the serviceman needs one to install a modification kit, so copy his if no other copy exists). Do not permit undocumented changes. These can make future troubleshooting impossible.

The diary should be kept close to the machine, and entries should be handwritten to avoid delays which breed forgetfulness. All entries should be dated and signed. Trouble entries should include not only the correction made (Replaced PC Board #6), but should also list the symptoms indicating the problem (would not print upper case). Only in this way can a diary facilitate your system's, and your own, transition to a golden old age.

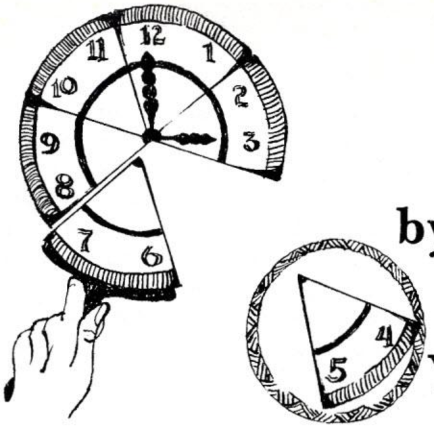
### Errata

Things not mentioned earlier since no one would fail to check them out first:

1. Plug pulled at wall.
2. Wall plug circuit failed (try test lamp in the same plug).
3. Switch off somewhere. Sometimes a component unit of a system will be energized from a central equipment bank, even though it may possess its own shutoff switch. Make sure someone hasn't flipped that other switch.
4. Fuse blown. Know where they are and how to check them.
5. Panel interlock switch. Some equipment has safety switches to kill live circuits if a panel door is opened. If the door is not fully closed. . . .
6. General cussedness. Just remember, machinery has no malice.

Now, when it's all said and done, one might add that these procedures aren't meant to displace the repairman from your life, nor that of your computer. But it should make for a happier *ménage à trois.* ▼

# Time Sharing on the Family Micro



by Barry Yarkon

Dad is in the den projecting December's family budget. Junior, working upstairs, is doing his chemistry homework. Mother's in the kitchen, proportionally enlarging her recipe for zucchini bread. And in the living room Uncle Ned, tonight's supper guest, keeps muttering, "one more time" as he plummets into lunar dust.

These people and activities may not be remarkable, but the fact that all four are using a single microcomputer—at the same time—is. The *time sharing* technique, originally developed for large computer systems, can finally be adapted to microcomputers with recently released time sharing (T/S) software. Now the cost of a microcomputer system can be divided among several users. Imagine the possibilities for schools, libraries, offices, and computer clubs.

### How Does It Work?

Time sharing was devised on large computer systems to exploit the large differences between the reaction speed of the person at a computer's terminal (typically, tenths of a second) and that of the interacting computer (thousandths of a second). While an operator pecks away at the keyboard, or, while a printer chugs along, the computer sits idly waiting for everyone to catch up. With time sharing, this excess capacity is utilized since the computer spends a short time with each user in rapid succession. Because each interval of time is so short (by normal human standards), an individual may not even be aware that the computer's attention comes and goes.

In order to demonstrate what can be done on a hobbyist microcomputer, let's use MITS' T/S BASIC. The software/hardware allows the computer to divide its time into slices of 16 2/3 milliseconds each. For this amount of time the computer devotes its attention to the user, temporarily saves his data, and then goes on to the next user or device. The computer can redirect its attention as many as sixty times a second. T/S BASIC can support up to eight users, and approximately four can be serviced simultaneously before a delayed response occurs at the terminals. Forced sharing prevents a single user from monopolizing the system with a problem that requires extensive processing.

In addition to allotting the computer's attention, time sharing also divides the computer's memory. T/S BASIC uses a "fixed partition" method of allocating memory. Each user is given a memory region which only he can access. The size of each user's share is determined at each start-up and need not be equally distributed among them. In our scenario, for example, Uncle Ned's game would require less memory space than Dad's calculations. The following Memory Map illustrates the allocation of bytes by "fixed partition":

T/S Disk BASIC Version 1.0

23,215 bytes

PARTITION ONE: 6484 bytes

PARTITION TWO: 6484 bytes

PARTITION THREE: 6484 bytes

PARTITION FOUR: 6484 bytes

TOTAL MEMORY = 49,152 bytes (48K)

Timesharing BASIC is an "interrupt driven" system. As the term implies, T/S BASIC was written to service a user's job on a demand schedule. This demand for attention is called an *interrupt*. Provided that the computer is conditioned to allow interruptions, it will suspend operations on its current job, save critical information about the job and its status for later resumption, and then spend some time processing the new job. It's like cutting-in on a dance at the prom. Slow processes, such as keyboard input, which generate interrupt requests only infrequently, will get less of

**Table 1**
**Hardware Structure of Altair T/S Disk BASIC (Version 1.0)**

| INTERRUPT PRIORITY | HARDWARE DEVICE | I/O DEVICE | PRECONFIGURATION |
|---|---|---|---|
| Level 0 | Disk controller boards (2) | 1-16 disk drives | Two drives |
| Level 1 | Real Time Clock | Interrupt timer | 16-2/3 millisecond intervals |
| Level 2 | 2SIO serial interface | 1 or 2 serial devices | Two devices at ports 16 & 18 decimal |
| Level 3 | 2SIO serial interface | 1 or 2 serial devices | Two devices at ports 20 & 22 |
| Level 4 | 2SIO serial interface | 1 or 2 serial devices | Two devices at ports 24 & 26 |
| Level 5 | 2SIO serial interface | 1 or 2 serial devices | Two devices at ports 28 & 30 |
| Level 6 | Line printer controller board | 1 character printer | Altair Q70 printer |
| Level 7 | Not used | None | |

the microcomputer's time than fast processes, i.e., disk input/output.

Conflicts between simultaneous interrupt requests from users are resolved by a hardware device called a *Vector Interrupt board.* Priorities for each user or device are established beforehand and are handled on a highest-priority-first basis. Don't be misled by the word "highest"; in T/S BASIC lower is better, so interrupt priority level 0 has more priority than interrupt level 7. Devices which require quick or frequent service are assigned "higher" priorities than those with less need; for example, the disk is at level 0 and the printer at level 6 or 7. In the Altair microcomputer scheme an add-on to the Vector Interrupt board, a *Real Time Clock* (RTC), monitors the 60-cycle AC power line and generates from it predetermined interrupt periods every 1/60 second.

### About Hardware

MITS recommends the following minimum time-sharing BASIC system (cassette based):

- an Altair 8800 microcomputer
- 32K bytes of RAM (max. 64K)
- Vector Interrupt board/RTC
- one 2SIO serial interface board for each two (2) users
- cassette interface and player
- T/S BASIC software (on cassette)

For their floppy disk-based version, Altair Timesharing Disk BASIC:

- an Altair 8800 microcomputer
- 32K bytes of RAM
- VI/RTC board
- one 2SIO board for each two users (max. eight users)
- one floppy disk controller and drive (up to sixteen drives)
- PROM bootstrap loader card
- Altair T/S BASIC software (on diskette)

The actual configuration is entirely dependent on the number of simultaneous users and devices required. To erect a four-terminal disk system, for example, you would require two 2SIO boards and, probably, two floppy disk drives. For each of the four users to have a good-sized partition in which to work would require approximately 48K bytes of RAM (49,152 bytes) would be necessary. The T/S Disk BASIC interpreter uses 23,215 bytes leaving 25,937 bytes of memory to be divided among the users. If each user were to have an equal share (6,484 bytes), his partition would yield :

| | |
|---|---|
| 1043 | system workspace |
| 100 | string space |
| 5341 | free program memory space |
| 6484 | total memory in partition |

So, to guarantee approximately 5K bytes to each of four users, 48K bytes of processor RAM are necessary. (As microcomputer software becomes more ambitious—and therefore memory hungry—the cost per byte of RAM hardware plummets downward.)

T/S BASIC can also support either MITS' Centronics or Qume character printers. A single printer can be shared among all users. In order to prevent conflicts, requests for printer service are automatically queued in software.

Table 1 details the hardware items described in the initialization dialog sequence which occurs each time the system is brought up. This is the initialization dialog for T/S Disk BASIC with four users:

```
ALTAIR T/S DISK BASIC V1.0 (6/16/77)
COPYRIGHT 1977 BY MITS INC.
HIGHEST ADDRESS IS 137777
RECONFIGURE (Y, N, L)? L' LIST
CURRENT I/O CONFIGURATION
LEVEL 0: 2 — DISK
LEVEL 1: TIMER
LEVEL 2: 2 — 16, 18
LEVEL 3: 2 — 20, 22
LEVEL 4: 2 — 24, 26
LEVEL 5: 2 — 28, 30
LEVEL 6: Q LINE PRINTER
LEVEL 7: NONE
RECONFIGURE (Y, N, L)?

                             N'NO
MEMORY SIZE? 49152
25937 BYTES AVAILABLE
NUMBER OF USERS? 4
TERMINAL ADDRESS? 16
REGION SIZE? 6484
TERMINAL ADDRESS? 18
REGION SIZE? 6484
TERMINAL ADDRESS? 20
REGION SIZE? 6484
TERMINAL ADDRESS? 22
REGION SIZE? 6484

1 BYTES UNUSED
DISK PASSWORD? ROM!

ALTAIR T/S DISK BASIC V1.0
COPYRIGHT 1977 BY MITS INC.
OK
```

Although T/S Disk BASIC is distributed with a configuration of two disk drives, eight terminals, and a Qume printer, you may reconfigure upon bringing it up and then save the new configuration on disk. Finally, a sign-on message is sent to each device followed by "OK".

A wide range of devices can be used as terminals. The author's system supported:

- Teleray 3811 CRT. An upper and lower case professional ASCII terminal with twenty-four lines of eighty characters and X-Y addressable cursor. Switched between 300/9600 baud.

- CT-1024. An upper case only TV terminal from SWTPC. Operating at 110/1200 baud.
- ASR-33 TeleType at 110 baud.
- GE Terminette 300 at 300 baud.

### About Software

Both versions, Altair T/S BASIC and Altair T/S Disk BASIC, allow each user access to most of the features of the well known Altair Extended BASIC.

To protect the integrity of the system from inadvertent user errors or encroachment, T/S BASIC disallows byte-oriented I/O functions such as: PEEK, POKE, INP, OUT, and WAIT. User-defined machine language subroutines (DEFUSE) are also disallowed—you must stay within your partition.

A password scheme provides further protection for disk based operations of reading and writing program files. At initialization a "disk password" of one to four characters is established. Each disk operation requires the password. Each file, then, can be treated in one of three ways:

*READ protect*—the file cannot be LOADed, RUN, KILLed, or reNAMEd without the correct password.

*WRITE protect*—the file cannot be KILLed or reNAMEd without the correct password.

*No protection*—password not required.

The password for an individual file can be easily changed before a file is SAVEd. Password protection is also provided for system commands such as MOUNT, UNLOAD, or DSKINI.

There is one shortcoming to full utilization of Altair Timesharing Disk BASIC: the current version (Version 1.0) does not handle *data files* on the disk. This capability is to be implemented in a future release. Many users who plan to exercise games and simulations, immediate calculations, program development, control systems, etc., will not be noticeably affected.

In the meantime, you can obtain partial relief from this problem by saving certain data types in a sequential *program file*. Such a "data file," masquerading as a program file, is illustrated in the two specimen programs that follow:

```
LIST'  This is the Data File as on Disk:


1100 REM  This is the Data-type Sequential File
          (It must be SAVEd in ASCII)

1110 DATA 243,"BEARING, Thrust",6,3.95
1120 DATA 177,"WASHER, Nylon",12,.25
1130 DATA 811,"TUBING,PVC",2,4.35
1140 DATA 904,"GEAR, Bronze",6,5.85
1150 DATA 622,"HOSING, Flex",1,12.50
1160 DATA 950,"CONNECTION",12,2.47
1170 REM End of DATA, 8.2.77

OK
```

```
LIST'  This is the Job File as on Disk:


100 REM  Simulation of Data Files in
         Altair T/S Disk BASIC, Ver. 1.0

110 PRINTCHR$(12): CLEAR 1024' Clr CRT
120 PRINT
    "-----------------------------------------------------------------------"
130 PRINT"Code:";TAB(10)"Item:","Quantity:","Price:","TOTAL:"
140 PRINT
    "-----------------------------------------------------------------------":
    PRINT:PRINT
150 READ I,I$,Q,P
160 PRINTI;TAB(10)I$;TAB(32)Q,P,;:PRINT USING "$##.##";Q*P
170 IF I=> 950THEN PRINT:PRINT
    "-----------------------------------------------------------------------"
    :PRINT:PRINT:STOP
180 GOTO 150
OK
```

Both files can be saved on disk. The application file (program) is LOADed into memory. Then the "data file" is MERGEd into it. Executing the combined file produces limited but acceptable results. Note, for example, the following command sequence and job printout:

```
'** Sequence of Commands to Do the Job:


OK

LOAD"JOBFILE",0
WORKING...
PASSWORD FOR FILE "JOBFILE "? ROM!
OK


MERGE"DATAFILE",0
WORKING...
PASSWORD FOR FILE "DATAFILE"? ROM!
OK


RUN'NOW RUN THE COMBINED FILES

-------------------------------------------------------------------
 Code:        Item:          Quantity:      Price:       TOTAL:
-------------------------------------------------------------------
  243    BEARING, Thrust     6            3.95         $23.70
  177    WASHER, Nylon       12           .25          $ 3.00
  811    TUBING, PVC         2            4.35         $ 8.70
  904    GEAR, Bronze        6            5.85         $35.10
  622    HOSING, Flex        1            12.5         $12.50
  950    CONNECTION          12           2.47         $29.64
-------------------------------------------------------------------


BREAK IN 170
OK
```

The economy of "less is more" now allows the cost of a microcomputer to be apportioned against several users, substantially reducing the price of full computing power to individuals. It also ends the "my turn next, my turn next!" syndrome; everyone gets to play—at the same time. ▼

When to lube? Usually every thirty to sixty days, to coincide with your cleaning. How much lube? When it starts to drip, stop oiling and start cleaning up. What kind of lube? Sewing machine oil for the oil points, power tool gearbox grease for the gears, and lithium-based greases for the cams. Avoid graphite, since it is messy and electrically conductive.

### Disasters

The author has already mentioned the affair with the rabbit. Perhaps you don't worry about bunnies in your data-processing environment, but other ghastly things can and do stalk you. Such as:

1. Equipment knocked over, dropped, or banged into.
2. Coffee, Coke, or milkshake spilled into a machine.
3. Roof leaks or sprinkler head opens, soaking the machine.
4. Fire.

*Undress the machine and check to see how deeply into the vitals the goop penetrated.*

Falls and bangs are most likely to happen to semiportable units such as terminals, cassette recorders, and video monitors. If the unit busts all to flinders, you have no problem. If, as is more likely, you dent it, bust the case, or just jar it badly, closer examination should reveal whether you can continue to operate or even turn it on for a test.

These eventualities call for undressing the machine, of course. Check for obvious and visible breakage. Then check for plugs and plug-in circuit boards jarred loose, components bent toward one another so that they touch and short, and bent or displaced mechanical parts. If all looks well, try moving mechanical parts through their cycles by hand to check for rubbing or interference. If all looks well and you can't wait to have it checked out by a serviceman, try putting the unit into service in such a manner that you can shut down at the first suspicion of danger. Observe the performance cautiously until you are convinced that normal operation occurs.

Foodstuffs spilled into a machine can be quite a mess. If you are fortunate enough to be there when it happens, first shut down as quickly as possible, in case blowing fuses have not already done that for you. Second, blot or wipe up as much of the mess as you can reach.

Third, undress the machine and check to see how deeply into the vitals the goop penetrated. If you are lucky, only the mechanical parts were caught. Very carefully wipe and wash away every trace of the substance, wipe dry, spray with WD-40 penetrating oil (protect printed circuit boards and electronic wiring from this), lubricate lube points, and have the serviceman check out the electronics.

If electronic components were soaked, wash the goop off with distilled water, treat mechanical parts as above, and get a serviceman to give it a controlled drying and checkout.

For the machine soaked by the roof leak or sprinkler head, cut it loose from all power, WD-40 it, and go for the professional dryout and checkout. Amateurs should never themselves try to dry out a system, since too much heat (even direct sun) can ruin components, and hidden water not eliminated can short or rust out the works.

In case of fire, if the fire is inside the equipment, call your insurance man and forget it. Almost any occurrence beyond minor smoking will generate problems far beyond the first aid described here.

If the fire is outside the equipment, a quick undressing can tell whether the heat got to the works. Unless insurance is involved here, you might try a cleanup and see if it gets you going. Even if it doesn't, the repairman will appreciate working on a clean machine.

### Dear Diary

Earlier, we discussed the importance of keeping all the documentation that came with your machine. Another set of documents just as critical is a running diary or log of everything that happens to the machine. While running time is nice to know, more important are cleaning and repair records (just how often did that capstan drive fail?), records of all modifications including corrected schematics (the serviceman needs one to install a modification kit, so copy his if no other copy exists). Do not permit undocumented changes. These can make future troubleshooting impossible.

The diary should be kept close to the machine, and entries should be handwritten to avoid delays which breed forgetfulness. All entries should be dated and signed. Trouble entries should include not only the correction made (Replaced PC Board #6), but should also list the symptoms indicating the problem (would not print upper case). Only in this way can a diary facilitate your system's, and your own, transition to a golden old age.

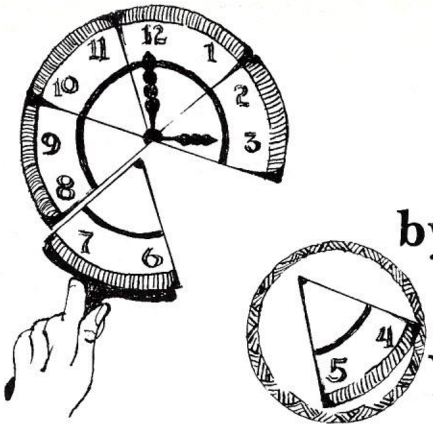
### Errata

Things not mentioned earlier since no one would fail to check them out first:

1. Plug pulled at wall.
2. Wall plug circuit failed (try test lamp in the same plug).
3. Switch off somewhere. Sometimes a component unit of a system will be energized from a central equipment bank, even though it may possess its own shutoff switch. Make sure someone hasn't flipped that other switch.
4. Fuse blown. Know where they are and how to check them.
5. Panel interlock switch. Some equipment has safety switches to kill live circuits if a panel door is opened. If the door is not fully closed. . . .
6. General cussedness. Just remember, machinery has no malice.

Now, when it's all said and done, one might add that these procedures aren't meant to displace the repairman from your life, nor that of your computer. But it should make for a happier *ménage à trois.* ▼

# Time Sharing on the Family Micro



## by Barry Yarkon

Dad is in the den projecting December's family budget. Junior, working upstairs, is doing his chemistry homework. Mother's in the kitchen, proportionally enlarging her recipe for zucchini bread. And in the living room Uncle Ned, tonight's supper guest, keeps muttering, "one more time" as he plummets into lunar dust.

These people and activities may not be remarkable, but the fact that all four are using a single microcomputer—at the same time—is. The *time sharing* technique, originally developed for large computer systems, can finally be adapted to microcomputers with recently released time sharing (T/S) software. Now the cost of a microcomputer system can be divided among several users. Imagine the possibilities for schools, libraries, offices, and computer clubs.

### How Does It Work?

Time sharing was devised on large computer systems to exploit the large differences between the reaction speed of the person at a computer's terminal (typically, tenths of a second) and that of the interacting computer (thousandths of a second). While an operator pecks away at the keyboard, or, while a printer chugs along, the computer sits idly waiting for everyone to catch up. With time sharing, this excess capacity is utilized since the computer spends a short time with each user in rapid succession. Because each interval of time is so short (by normal human standards), an individual may not even be aware that the computer's attention comes and goes.

In order to demonstrate what can be done on a hobbyist microcomputer, let's use MITS' T/S BASIC. The software/hardware allows the computer to divide its time into slices of 16 2/3 milliseconds each. For this amount of time the computer devotes its attention to the user, temporarily saves his data, and then goes on to the next user or device. The computer can redirect its attention as many as sixty times a second. T/S BASIC can support up to eight users, and approximately four can be serviced simultaneously before a delayed response occurs at the terminals. Forced sharing prevents a single user from monopolizing the system with a problem that requires extensive processing.

In addition to allotting the computer's attention, time sharing also divides the computer's memory. T/S BASIC uses a "fixed partition" method of allocating memory. Each user is given a memory region which only he can access. The size of each user's share is determined at each start-up and need not be equally distributed among them. In our scenario, for example, Uncle Ned's game would require less memory space than Dad's calculations. The following Memory Map illustrates the allocation of bytes by "fixed partition":

T/S Disk BASIC Version 1.0

23,215 bytes

PARTITION ONE: 6484 bytes

PARTITION TWO: 6484 bytes

PARTITION THREE: 6484 bytes

PARTITION FOUR: 6484 bytes

TOTAL MEMORY = 49,152 bytes (48K)

Timesharing BASIC is an "interrupt driven" system. As the term implies, T/S BASIC was written to service a user's job on a demand schedule. This demand for attention is called an *interrupt*. Provided that the computer is conditioned to allow interruptions, it will suspend operations on its current job, save critical information about the job and its status for later resumption, and then spend some time processing the new job. It's like cutting-in on a dance at the prom. Slow processes, such as keyboard input, which generate interrupt requests only infrequently, will get less of

**Table 1**
**Hardware Structure of Altair T/S Disk BASIC (Version 1.0)**

| INTERRUPT PRIORITY | HARDWARE DEVICE | I/O DEVICE | PRECONFIGURATION |
|---|---|---|---|
| Level 0 | Disk controller boards (2) | 1-16 disk drives | Two drives |
| Level 1 | Real Time Clock | Interrupt timer | 16-2/3 millisecond intervals |
| Level 2 | 2SIO serial interface | 1 or 2 serial devices | Two devices at ports 16 & 18 decimal |
| Level 3 | 2SIO serial interface | 1 or 2 serial devices | Two devices at ports 20 & 22 |
| Level 4 | 2SIO serial interface | 1 or 2 serial devices | Two devices at ports 24 & 26 |
| Level 5 | 2SIO serial interface | 1 or 2 serial devices | Two devices at ports 28 & 30 |
| Level 6 | Line printer controller board | 1 character printer | Altair Q70 printer |
| Level 7 | Not used | None | |

the microcomputer's time than fast processes, i.e., disk input/output.

Conflicts between simultaneous interrupt requests from users are resolved by a hardware device called a *Vector Interrupt board.* Priorities for each user or device are established beforehand and are handled on a highest-priority-first basis. Don't be misled by the word "highest"; in T/S BASIC lower is better, so interrupt priority level 0 has more priority than interrupt level 7. Devices which require quick or frequent service are assigned "higher" priorities than those with less need; for example, the disk is at level 0 and the printer at level 6 or 7. In the Altair microcomputer scheme an add-on to the Vector Interrupt board, a *Real Time Clock* (RTC), monitors the 60-cycle AC power line and generates from it predetermined interrupt periods every 1/60 second.

### About Hardware

MITS recommends the following minimum timesharing BASIC system (cassette based):

- an Altair 8800 microcomputer
- 32K bytes of RAM (max. 64K)
- Vector Interrupt board/RTC
- one 2SIO serial interface board for each two (2) users
- cassette interface and player
- T/S BASIC software (on cassette)

For their floppy disk-based version, Altair Timesharing Disk BASIC:

- an Altair 8800 microcomputer
- 32K bytes of RAM
- VI/RTC board
- one 2SIO board for each two users (max. eight users)
- one floppy disk controller and drive (up to sixteen drives)
- PROM bootstrap loader card
- Altair T/S BASIC software (on diskette)

The actual configuration is entirely dependent on the number of simultaneous users and devices required. To erect a four-terminal disk system, for example, you would require two 2SIO boards and, probably, two floppy disk drives. For each of the four users to have a good-sized partition in which to work would require approximately 48K bytes of RAM (49,152 bytes) would be necessary. The T/S Disk BASIC interpreter uses 23,215 bytes leaving 25,937 bytes of memory to be divided among the users. If each user were to have an equal share (6,484 bytes), his partition would yield:

| | |
|---|---|
| 1043 | system workspace |
| 100 | string space |
| 5341 | free program memory space |
| 6484 | total memory in partition |

So, to guarantee approximately 5K bytes to each of four users, 48K bytes of processor RAM are necessary. (As microcomputer software becomes more ambitious—and therefore memory hungry—the cost per byte of RAM hardware plummets downward.)

T/S BASIC can also support either MITS' Centronics or Qume character printers. A single printer can be shared among all users. In order to prevent conflicts, requests for printer service are automatically queued in software.

Table 1 details the hardware items described in the initialization dialog sequence which occurs each time the system is brought up. This is the initialization dialog for T/S Disk BASIC with four users:

```
ALTAIR T/S DISK BASIC V1.0 (6/16/77)
COPYRIGHT 1977 BY MITS INC.
HIGHEST ADDRESS IS 137777
RECONFIGURE (Y, N, L)? L' LIST
CURRENT I/O CONFIGURATION
LEVEL 0: 2 — DISK
LEVEL 1: TIMER
LEVEL 2: 2 — 16, 18
LEVEL 3: 2 — 20, 22
LEVEL 4: 2 — 24, 26
LEVEL 5: 2 — 28, 30
LEVEL 6: Q LINE PRINTER
LEVEL 7: NONE
RECONFIGURE (Y, N, L)?

                                   N'NO
MEMORY SIZE? 49152
25937 BYTES AVAILABLE
NUMBER OF USERS? 4
TERMINAL ADDRESS? 16
REGION SIZE? 6484
TERMINAL ADDRESS? 18
REGION SIZE? 6484
TERMINAL ADDRESS? 20
REGION SIZE? 6484
TERMINAL ADDRESS? 22
REGION SIZE? 6484

1 BYTES UNUSED
DISK PASSWORD? ROM!

ALTAIR T/S DISK BASIC V1.0
COPYRIGHT 1977 BY MITS INC.
OK
```

Although T/S Disk BASIC is distributed with a configuration of two disk drives, eight terminals, and a Qume printer, you may reconfigure upon bringing it up and then save the new configuration on disk. Finally, a sign-on message is sent to each device followed by "OK".

A wide range of devices can be used as terminals. The author's system supported:

- Teleray 3811 CRT. An upper and lower case professional ASCII terminal with twenty-four lines of eighty characters and X-Y addressable cursor. Switched between 300/9600 baud.

- CT-1024. An upper case only TV terminal from SWTPC. Operating at 110/1200 baud.

- ASR-33 TeleType at 110 baud.

- GE Terminette 300 at 300 baud.

### About Software

Both versions, Altair T/S BASIC and Altair T/S Disk BASIC, allow each user access to most of the features of the well known Altair Extended BASIC.

To protect the integrity of the system from inadvertent user errors or encroachment, T/S BASIC disallows byte-oriented I/O functions such as: PEEK, POKE, INP, OUT, and WAIT. User-defined machine language subroutines (DEFUSE) are also disallowed—you must stay within your partition.

A password scheme provides further protection for disk based operations of reading and writing program files. At initialization a "disk password" of one to four characters is established. Each disk operation requires the password. Each file, then, can be treated in one of three ways:

*READ protect*—the file cannot be LOADed, RUN, KILLed, or reNAMEd without the correct password.

*WRITE protect*—the file cannot be KILLed or reNAMEd without the correct password.

*No protection*—password not required.

The password for an individual file can be easily changed before a file is SAVEd. Password protection is also provided for system commands such as MOUNT, UNLOAD, or DSKINI.

There is one shortcoming to full utilization of Altair Timesharing Disk BASIC: the current version (Version 1.0) does not handle *data files* on the disk. This capability is to be implemented in a future release. Many users who plan to exercise games and simulations, immediate calculations, program development, control systems, etc., will not be noticeably affected.

In the meantime, you can obtain partial relief from this problem by saving certain data types in a sequential *program file*. Such a "data file," masquerading as a program file, is illustrated in the two specimen programs that follow:

```
LIST'   This is the Data File as on Disk:

1100 REM  This is the Data-type Sequential File
          (It must be SAVEd in ASCII)

1110 DATA 243,"BEARING, Thrust",6,3.95
1120 DATA 177,"WASHER, Nylon",12,.25
1130 DATA 811,"TUBING,PVC",2,4.35
1140 DATA 904,"GEAR, Bronze",6,5.85
1150 DATA 622,"HOSING, Flex",1,12.50
1160 DATA 950,"CONNECTION",12,2.47
1170 REM End of DATA, 8.2.77

OK
```

```
LIST'   This is the Job File as on Disk:

100 REM  Simulation of Data Files in
         Altair T/S Disk BASIC, Ver. 1.0

110 PRINTCHR$(12): CLEAR 1024' Clr CRT
120 PRINT
    "-----------------------------------------------------------------------"
130 PRINT"Code:";TAB(10)"Item:","Quantity:","Price:","TOTAL:"
140 PRINT
    "-----------------------------------------------------------------------";
    PRINT:PRINT
150 READ I,I$,Q,P
160 PRINTI;TAB(10)I$;TAB(32)Q,P,;:PRINT USING "$##.##";Q*P
170 IF I=> 950THEN PRINT:PRINT
    "-----------------------------------------------------------------------"
    :PRINT:PRINT:STOP
180 GOTO 150
OK
```

Both files can be saved on disk. The application file (program) is LOADed into memory. Then the "data file" is MERGEd into it. Executing the combined file produces limited but acceptable results. Note, for example, the following command sequence and job printout:

```
 '** Sequence of Commands to Do the Job:


OK

LOAD"JOBFILE",0
WORKING...
PASSWORD FOR FILE "JOBFILE "? ROM!
OK


MERGE"DATAFILE",0
WORKING...
PASSWORD FOR FILE "DATAFILE"? ROM!
OK


RUN 'NOW RUN THE COMBINED FILES

______________________________________________________________
 Code:         Item:          Quantity:     Price:     TOTAL:
______________________________________________________________
  243   BEARING, Thrust        6           3.95       $23.70
  177   WASHER, Nylon          12          .25        $ 3.00
  811   TUBING, PVC            2           4.35       $ 8.70
  904   GEAR, Bronze           6           5.85       $35.10
  622   HOSING, Flex           1           12.5       $12.50
  950   CONNECTION             12          2.47       $29.64
______________________________________________________________


BREAK IN 170
OK
```

The economy of "less is more" now allows the cost of a microcomputer to be apportioned against several users, substantially reducing the price of full computing power to individuals. It also ends the "my turn next, my turn next!" syndrome; everyone gets to play—at the same time. ▼

small flying devices shaped like water drops darted from the sky. Their blue glow and peeping *dwee-dwee-dwee* gave him much warning. He had time—to dodge their fire bolts and to arm himself with stones. The Strength gave spring to his thick legs. He catapulted the stones from them so hard and true that four of the devices exploded into smoking crumbles. The last fled and he scraped his haunches toward its cowardly flight. Then he plunged down into the wilderness and safety.

**_Sweeps of natural blue foliage shielding him from sight suddenly writhed into blackness and died in shrivels._**

In the deepest thicket, ten days running journey from any village—a net! It fell from a tall hex tree snarling him in its metal mesh. He knew it came from G.O.D., so rage boiled up in him. The mesh was strong, but at last he broke two strands, slipped out, and away.

He began to run, then, through the wilderness, for he knew there was no safety from G.O.D. anywhere. Projectiles crashed down from the blue sky, spewing green vapors. Sweeps of natural blue foliage shielding him from sight suddenly writhed into blackness and died in shrivels. Winged lizards and furry bushers sniffed the vapor and turned legs up, oozing blood from tiny nostrils.

He ran on, calling in vain on the Strength from the depths of his despair.

How long he was hunted he could not guess. He could not remember when he was last at rest. The suns rose and sank, the moons followed. He fled, but could not escape the eyes of G.O.D., the machine that thought.

There came a hot midday when great searching engines pushed through the jungle behind him. He knew that he was being driven. Too late he saw that he ran down a narrowing peninsula bounded by an ocean, above which pulsing discs waited to discharge their lethal bolts upon him should he swim, unable to dodge.

The land rose toward a height. There an ancient burrow was crowned by the crumbling parapets and ramps of an antique Digger king's stronghold. Luuum hurdled the wall of crumbling stones. Standing alone in the flattened clay freespace was a young Digger woman gowned as in the days of the ancients. She beckoned to him. Her eyes burned with zeal. "Come," she droned. "Sing to me of the Strength."

Perhaps this was a trap of G.O.D. A trick. Yet she was real and he was so weary of flight....

He began his song.

Her eyes widened as her membranes drank its beauty.

Luuum heard a rushing from the sky. But whatever was to be, the female was a fellow Digger. He rushed to her side to offer strength and comfort.

The rushing's pitch brightened. A dome slammed down, fitting grooves he hadn't seen, sealing. They were encased! He dove to dig. Beneath a great toe's depth of clay—plates from which his nails struck sparks, but gained no purchase.

The air was rushing from the dome!

His eyes turned to the female. Her eyes were rapt. Even as she died, her paws each swept out a head-sized circle before her chest. "I sacrifice... my life to G.O.D. to help its sublime ends. May I... be blessed...." Circles, even as she fell. Circles... circles....

Horb made the G.O.D. circle in greeting Dormus. "G.O.D.-Is-One," he said. His fellow Analyst returned gesture and greeting.

"I don't see why G.O.D. wanted anals on that burrower, Horb," the feathered being's speakunit chirped. He jerked a stubby wing toward the Analyticon in which each atom of the Digger's corpse was being examined.

"G.O.D.-knows."

"So it's said. But why...."

Horb waved a printon narrative. "A few small things that advocates of rude dieties—the "Strength" in this case—generally don't do." He tapped the printon. "Like crush a homing node in his paw. The fracture point for its cryston shell was 2000 kilos per square centimeter."

"What?"

"And paddle across a lake 1800 kilometers wide. Dodge laserlashes as if they were spears. Tear net cables with his paws."

"Impossible!"

"How about catapulting ordinary stones at velocities approaching twice the speed of sound and hitting moving killerettes—hard. And running...."

"No living being could do those things," Dormus' speakunit chirped.

"None could."

"The Analyticon beeped and Horb slipped out the fresh printon. He nodded. "I guessed this. That creature was a... device." His three-eye cluster sought his colleague's befeathered visage. "An oxypowered device... such as the Analyticon has never experienced."

"But G.O.D.-knows-all."

"Not this. That creature couldn't have been anything—according to this data—but a part of a computer."

Dormus' wings brushed his head in confusion. G.O.D. had not taught him how to deal with this. "Why... what... was he... it... doing on this world?"

"I think I know."

"What?"

"The same thing the Units were. Carrying word of *his* computer. *One* device sent to do the same job G.O.D. sent thousands to do."

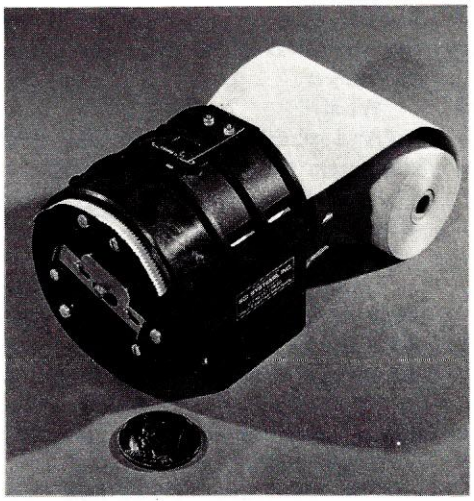
Dormus babbled, fluttering up and falling back to claws, utterly shaken.

"Don't you see what's happened?" Horb waved his paw from behind his head forward. "G.O.D. has come from across ten thousand stars to this world. This "Luuum's" Strength has come forward from the other direction. And the two have met here, accidentally, on this ball of mud."

"But G.O.D. *destroyed* the alien device."

Horb nodded. "The Strength will probably take action. By what means, and how many lifetimes it'll take to send that action across the stars, even G.O.D. doesn't know, but...."

The twin suns winked out! Horb knew they had not gone out, but had been *put* out—forever. From unseen generators of celestial power, waves of sound drove Horb and Dormus flat to the earth and ripped their auditories till fluids came. The words pounded down like mountains: *"How* dare *you harm one of* my *components!"* ▼



# THE WORDSLINGER 2200 Characters Per Second

## by Stuart Dambrot

How many of us have had haunting dreams about a machine that could print all the articles in this magazine in less than three minutes? Or give a hardcopy of a full twenty-four-line by eighty-character display in 3/4 of a second?

Stop dreaming. Such a device is now available to the home computerist. It is SCI System's Series 1100 Rotary Printer, a high-speed device with the following features:

- A printing rate of 2200 characters per second
- Dimensions of four by five by nine inches—including paper supply
- Line lengths limited only by the amount of paper on the roll
- Extreme quiet—a typewriter is typically six times as loud
- Complete interface circuitry
- High reliability, with life expectancies of twenty-five million characters for the print head(s) and *eight billion* for the entire mechanism
- Almost maintenance-free care—there are only nine moving parts, and the print head can be replaced in ten seconds
- Low cost—from $300 (if you buy in quantity) to about $1000 for an individual unit
- Probably the lowest cost-to-performance ratio in the industry

The SCI Rotary Printer uses three-layered electrosensitive paper: the base paper has a black pigment coating, on which an aluminum layer with a thickness of precisely 0.000001 inch (!) is applied. Printing is impactless. An electrical impulse is sent to any of five wires, or styli, in the print head; the current in these wires vaporizes the aluminum, exposing a dot of pigment. (See Figure 1.)

The character format is a five-by-seven dot-matrix. A character is printed by sending pulses to the right wires at the right time, as the styli—which are on a rotating disk—brush along the paper. The letter "A," for example, would be printed as shown in Figure 2.

Material is printed by column; memory access must be arranged in a way that allows data to be sent to the printer by column. A maximum of twenty-four lines can be printed; the number of columns is limited only by the length of paper. Thus, the first twenty-four characters transmitted constitute the first column. The Series 1100 Printer has three printing heads, allowing three columns to be printed per rotation. If the message does not require twenty-four lines, space characters (Hex 020) must be transmitted such that each column has twenty-four character positions.

Data fed to the printer first enters a 40-character FIFO buffer. ASCII data (from a 64- or 128-character set)

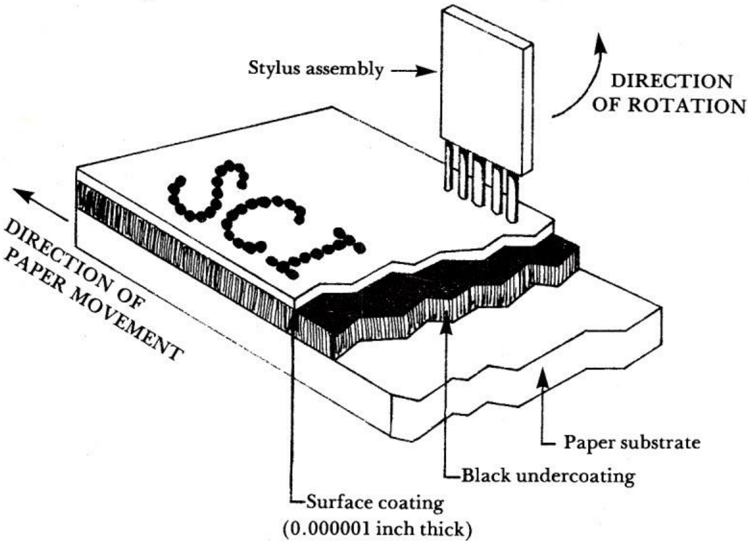


**Figure 1**

**Rotary Printing Method Utilizing Electrosensitive Paper**

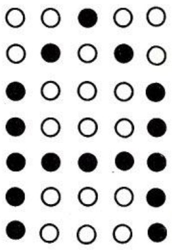


**Figure 2**

**The Character *A* as Realized on a Five-by-Seven Dot Matrix Forma**

is clocked from the FIFO buffer one symbol at a time. The printer begins operation upon request (e.g., "PRINT"). Execution can be halted either by command or when the on-board circuitry senses a "FIFO-empty" condition.

In operation, a drive roller continuously pushes paper through the unit. Character data is electronically converted to pulses which are sent to the styli; the appropriate symbol is then printed. A coded disk on the same shaft as the rotating print head identifies the character being printed. A photocell senses the position of the slots on the disk; this information is then used to time the print pulses, assuring constant-sized characters regardless of motor speed. The printer may be interfaced as shown in Figure 3.

The one drawback to the Rotary Printer is its unusual paper size: the rolls are four inches by two hundred feet. Future SCI printers, however, will be able to handle a variety of paper shapes and sizes—in addition to other functions such as a print rate of 4000 characters per second and graphics capabilities.

There are many potential applications of this low-cost, high-speed printer. Its small size allows it to be placed *inside* a CRT terminal. With its high print rate, displayed data can be hardcopied without creating a lag in usable processing time. One could easily install an SCI Rotary Printer in an automobile; under microprocessor control, records of gas mileage and elapsed time could be automatically hardcopied. One can envision innovations such as an "electronic newspaper": all available information would be displayed on a CRT device; as much material as a typical Sunday *New York Times* could be printed in about ten minutes. (Here a different paper format would obviously be necessary.)

SCI Rotary Printers are available in two basic configurations. The Model 1100 comes with the interface circuit and consists of a printing mechanism and the control card. It requires a software program to format the data transmission and control the printer that typically is forty to fifty instructions long. A model 1110 is completely self contained: it includes the printer mechanism, control card, a 24 x 136 character page memory, power supply, and a protective enclosure. The 1110-P version has a parallel interface which emulates conventional printers, and the 1110-S has an RS-232-C and a twenty to forty milliampere TTY current loop interface. Reformatting of data from lines to columns is provided automatically within the Model 1110, and therefore does not require a special software program for its application.

With its high speed and reliability, its low cost and compact size, the SCI Series 1100 Rotary Printer is definitely something to look into, so check it out. For further information, you can write to SCI Systems, Inc., 8600 South Memorial Parkway, Huntsville, Alabama 35802. ▼

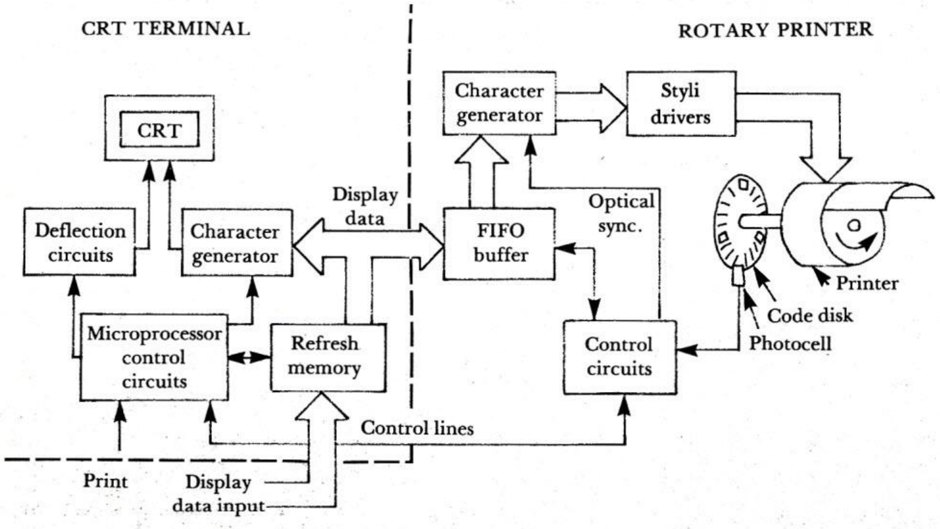


**Figure 3**

**Interface Design**

**ROMtutorial ROMtutorial**

*Interface:* The boundary between different sections of a computer, or between the computer and the outside world. The circuitry which crosses the boundaries is also called the interface.

*Hex:* Hexadecimal. A hex digit is a member of the set of sixteen digits: 0 through 9 and A through F, where A-F represent the decimal numbers 10-15. For example, hexadecimal A3E = [ (10×16 ) + (3×16) + (14) ]

$$= [2560 + 48 + 14]$$
$$= 2622.$$

*Buffer:* A temporary storage area which is used to equalize or balance different operating speeds. For example, a buffer can be used between a slow input device, such as a typewriter, and the main computer which operates at a very high speed.

*FIFO:* First-in-first-out. A method used in taking items from a list. An example of a FIFO process would be the way people go through the line at a cash register.

*Clocking:* Regular pulses sent by a time-keeping device within the computer system that schedules computer operations.

*Refresh memory:* A memory device which transmits display data to the CRT at short, regular intervals. This is necessary, because as the electron beam scans the CRT screen, the glow from the phosphor coating rapidly fades. The deflection circuit must be "reminded" where to direct the beam if the data is to be continuously displayed.

*Emulate:* The ability of one system to imitate another, with the imitating system accepting the same data and programming and achieving the same results as imitated system, but possibly at a different rate of processing.

*Parallel interface:* A device that accepts data when all bits of a particular character are being transmitted simultaneously.

*RS-232-C:* Identifies a standard interface.

*TTY current loop:* A method of transmitting the bits of a character serially, or one at a time. TTY stands for Teletype, a device with a keyboard and printer; a TTY and printer can send each other data, and this is accomplished through the current loop.

# LIGHT FANTASTIC
# The Kinetic Sculpture of Michael Mayock

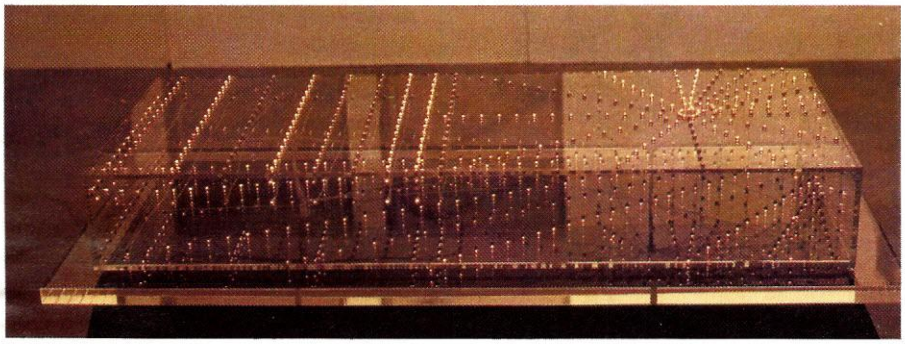
## by Tom Moldvay and Lawrence Schick

Has runaway technology thrust us totally into a dehumanizing machine age that sacrifices aesthetics to functionalism? One artist, Mike Mayock, is turning the products of mass technology around and transforming them, perhaps in spite of themselves, into works of art. He creates kinetic sculpture with solid-state electronics and binary computer logic and flashing, ever changing lights.

Mike is a self-taught electronics engineer. He made his first blinking light array when he was about eight years old, the age at which he also tried to fix his parents' television—fairly successfully, apparently, since they used it for three or four years afterwards. He just started connecting wires, without trying to read anything because everything written on the subject at the time was for technicians and he didn't understand what they were talking about. He fixed radios by trying different tubes, rummaging around for things until sooner or later he hit on something that worked. He also got burned a lot.

> *Sometimes a cheap little capacitor you can never depend on is the reason why the whole thing works.*

Mike learned about computers the way he learned about radios and television sets. And when he figured out, on one particular job, how to make forty-seven stock quotation machines in a day rather than in a week, he was kicked upstairs to Research and Development to work right inside a computer. He made elaborate mini-computer setups for taking care of a whole building—office routines, the machines, the works—interfacing an entire environment into a machine.


*For the Dead in Space*

Maybe being self-taught allowed Mike to break through the sometimes iconoclastic rules of engineering when he turned to electronic art. "I break all the rules," Mike says, "because I don't know them."

When you look at Mike's sculpture, what you see first is the flashing lights. As you follow the light sequences, you begin to realize that the patterns are not repetitive, as you at first supposed. Your attention shifts as the diodes blink on and off and random details of form are emphasized.

Mike's sculptures are built around a central, very simple idea of oscillators and numbers—numbers in a digital train. He uses PROMs, ROMs, EPROMs, and scratch pad memories. He counts numbers and he runs numbers together and over each other, and then he outputs them in a basic times-ten output. Everything comes in and out of ten little ports, but they're time-shared with other inputs.

Mike is rarely able to plan the final appearance of a piece ahead of time.

Photographed by Michael Mayock

The wires all over the place, the eight million pieces all over the basement have a way of taking on a life of their own.

"I guess the electronics, the actual how-it's-going-to-work, come after the idea," Mike admits, "and sometimes I can't do what I want to do. I find myself doing something that simply isn't allowable in technical terms."

One of Mike's most complex pieces was a depiction of Frederik Pohl's *Man Plus* astronaut, Roger Torraway, who was completely remade to survive on the planet Mars by replacing most of his body with mechanical and electronic hardware, directly interfacing his nervous system with highly sophisticated computers, and powering his new body with solar energy received through two sensitive shoulder "wings." *Man Plus* took five light systems—brain, skin, wings, nose-mouth, and eyes—all controlled with programmed patterns and arrangements and timings, and two special little circuits that constitute something of a trade secret. Then there are a couple of simple divide-bys, like divide-by sixteens, and tens, and three master oscillators. Mike arranges the hookups of the divide-bys to pick up the amount of each frequency that he wants. Normally he gets a system frequency and a clock frequency for each system. And they vary against one another. One frequency will shift everything up and down mathematically in terms of speed of movement while another one changes the intensity relative to that speed of movement. There are sixty-four bytes, each with an eight-bit word, in the PROM memory, and there are several PROMs, so Mike can do a variety of things.

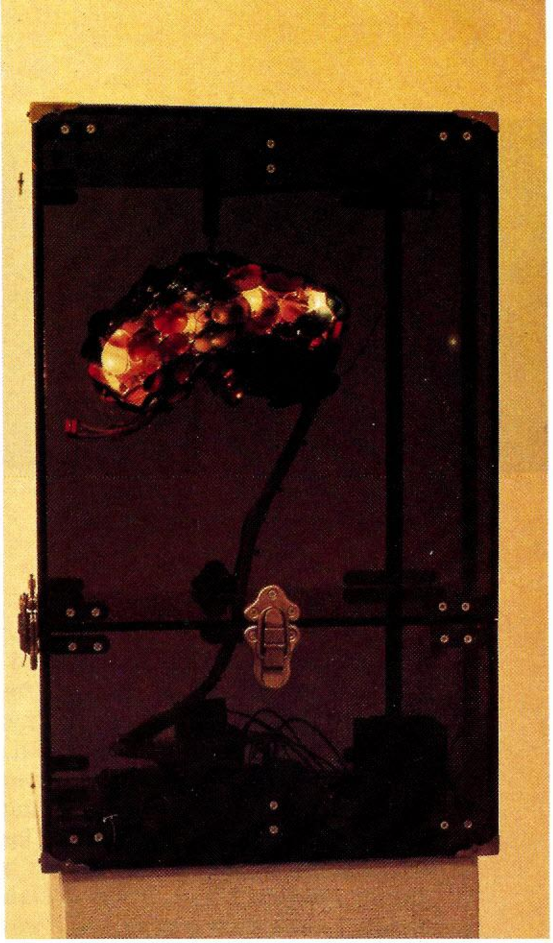
*All this digital stuff—it's a trap, because the numbers work so nicely.*

The brain of *Man Plus* was designed on a technical Christmas tree unfolded in the hollow interior. A sweep rate pattern of amber, red, and green diodes gives a sweeping and whirling effect like a vortex, almost as if information were being sucked in and processed. Even though the skin is porous you can't always see the diodes. What you do see is a lot of pulsating.

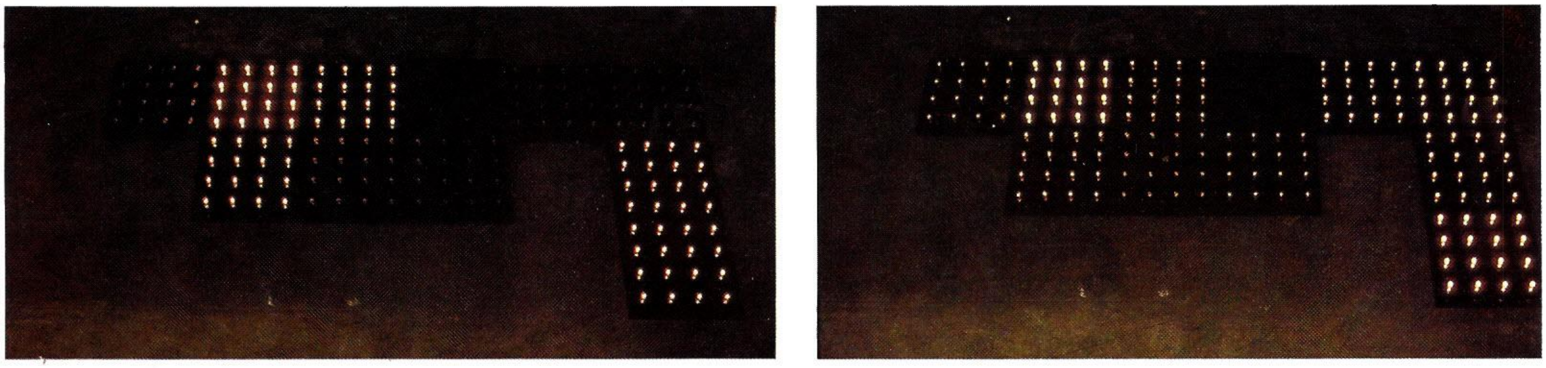
The astronaut's eyes were made of calculator readouts, seven in each eye, arranged in a circular convex lens. In the inputs to the eye system, two different frequencies are run across each other, one through the anode and the other through the cathode input. Both frequencies operate each separate light in the eyes at different times. There's a main frequency, and then Mike adds a frequency from a totally different oscillator, say, from one of those little divide-by chains before the outputs in the AC relays are reached, for a nice random element. "You get beat frequencies," Mike explains, "because you've got three master oscillators and you use a wide-tolerance capacitor that you know is going to vary a lot with heat. That one cheap little capacitor whose value you can never depend on at any given time is one of the keys to the reason the whole thing works."

But those eyes of *Man Plus* gave Mike some trouble. To make a number in the output, usually all he has to do is supply a logic 1 or 0, but the astronaut's eyes required two separate sets of inputs. "There's a mistake that one always makes," Mike says. "You figure, well, we've got all this powerful digital stuff, we'll just slap it in there with a few peripheral devices like power drivers, and we won't have any trouble.... WRONG! It's a trap, because the numbers work so nicely. Digital is so very far removed from electronics per se, you stop thinking electronically any more. You start thinking in terms of what this gate is going to do, what this arrangement of numbers in space is going to look like, and you forget little basic things.

*You start thinking what this arrangement of numbers in space is going to look like, and you forget little basic things.*


*The Brain*

"What happened was, I forgot about that current business. I got all the materials, all the bills were outstanding, and I put the son of a gun all together, and it all worked, and I felt really good. Then I built its own power supply, and it didn't work. I went, 'Ulp!' I tried using some big filter capacitors, like 75,000 microfarads, what they call garbage cans. Four of them held the line quiet enough for it to operate. There was so much current that it took one hell of a regulation system to handle the output. Otherwise the current would drop down and start drawing, and it really hit that power supply hard. When that happens, the power supply says, 'Uh-oh, overload.' It's not really an overload, but because of the peripheral drivers for the eyes (which turn on and off a five-volt faucet), the current comes in fast and hard. Those peripheral drivers are very versatile, I use them all the time, even though they hog a juicy amount of current. They just have this one problem: they shut off a five-volt stream for ten nanoseconds, so for ten nanoseconds that thing is dead, plonk! Three hundred mils moving that fast is a horrendous spike, because inductively it looks like a great deal more.



*Linear Cloud*

The power supply just goes bananas and says 'No!' That's what happened.

"I figured I'd have to put about six million of these capacitors in there for it to work, and the more I looked at it, the less I liked it. Fortunately, I had this $380 high-volt power supply lying around. It's really super, puts out a lot of beautifully regulated current, but it was the only one I had. It was my lab supply. I knew there was no way I was going to be able to hand-build a power supply, in the time I had, that would handle what I wanted it to handle. So I put my lab supply in there, and it worked just fine. But that's the kind of thing that happens, because sometimes when I'm doing these things I forget about the real physical world requirements."

On the other hand, of course, sometimes things that couldn't conceivably work—do. The people who supplied Mike with the parts to build *Linear Cloud* said it was an impossible piece. Mike blinks the lights on and off with a solid-state relay that shouldn't function at the high frequency he asks of it. Why? Because, Mike says, "nobody pushed the relay concept beyond turning on and off. Nobody dreamed of pushing that thing a hundred thousand times a second, because no one ever did that to a relay."

*For the Dead In Space* was another impossible. That one burned twenty-four hours a day for four months straight without failing once. It should have burned out.

Mike plans to overcome another can't-be-done in the future. He hopes to wed sound and light in his work, to connect what he hears the computer doing and what the lights do. People separate what they see and what they hear, and Mike has to find a way of putting the two together again. Which, as he says, is not easy to do. ▼

ROM
COMPUTER APPLICATIONS FOR LIVING

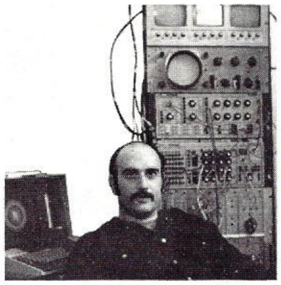
For advertising information, please contact the
publisher, Erik Sandberg-Diment.
ROM Publications Corp.
Route 97
Hampton, CT 06247
Tel. 203-455-9591

## Solution to last month's PROMpuzzle

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S | E | E | K | ■ | ■ | S | W | I | S | S | ■ | ■ | L | I | N | K |
| I | N | T | S | ■ | ■ | T | E | N | E | T | ■ | P | A | R | E | E |
| A | D | O | R | N | ■ | A | D | D | ■ | R | O | A | D | ■ | W | E |
| M | O | N | ■ | A | I | R | S | ■ | S | E | N | T | ■ | H | E | N |
| ■ | ■ | S | H | I | F | T | ■ | R | O | S | E | ■ | S | I | R | ■ |
| ■ | ■ | ■ | A | L | S | ■ | S | E | T | S | ■ | T | O | N | ■ | ■ |
| A | C | E | R | S | ■ | S | A | S | S | ■ | R | E | A | D | I | N |
| C | H | I | P | ■ | H | A | R | I | ■ | M | A | S | K | ■ | D | E |
| R | E | S | ■ | R | E | G | I | S | T | E | R | S | ■ | L | E | O |
| E | S | ■ | S | O | L | E | ■ | T | E | L | E | ■ | K | L | A | N |
| S | T | E | P | U | P | ■ | W | O | N | T | ■ | G | I | L | L | S |
| ■ | ■ | C | U | T | ■ | M | A | R | S | ■ | G | A | L | ■ | ■ | ■ |
| ■ | P | H | D | ■ | J | O | T | S | ■ | F | A | M | O | S | ■ | ■ |
| M | O | O | ■ | F | A | D | S | ■ | P | A | G | E | ■ | H | S | M |
| I | I | ■ | S | A | B | U | ■ | P | O | I | ■ | S | C | A | L | E |
| S | N | E | E | R | ■ | L | A | S | E | R | ■ | ■ | I | R | A | N |
| S | T | A | R | ■ | ■ | E | D | I | T | S | ■ | ■ | O | E | M | S |

futuROMa

# THE EYE OF THE *DAPHNIA*



### by Bill Etra

What does the *Daphnia* brain do in terms of electronic behavior? What I'm really asking—and suggesting—is, why can't we simulate complex organisms by electronic methods? Now it's understood that there are biochemical reactions going on in living organisms, and that these reactions are partly electrical or electronic in nature. It's also understood that our ability to study these organisms is limited. But two methods of bioelectronic interest are being used in the study and creation of modern electronic chips.

The scanning electron microscope, although it deals primarily with dead material—material which does not have any electronic charge going through it other than the scanning beam—is also capable of scanning minute particles of living cells and tracing out things such as nerve bundles. And IBM is using, in their electron-beam recording of chips, a process that is almost the exact reverse; they take an electron beam and disrupt the pattern in the material or impregnate the material to cause conductivity or hyperconductivity.

What if we were to scan a living organism section by section using such methods?

My suggestion is that if you were to read with one hand and copy with the other hand, instead of the standard electric model of the human brain being a warehouse something like four blocks by four blocks by sixteen stories high, you might be able to recreate a fair model of it in an electronic device approaching a scale of one to one, or at least no greater than ten to one. This would begin to bring biological organisms within the reach of computer models or electronic models that could be tested.

True, you might not be able to trace all the neurological signal endings. And in fact all of them may not be significant. But *some* of the biological interaction should be modelable in reasonable scale by electron-beam recording onto chips. What you'd get would be a chip which is a model of part of a living organism and on which you could test your theories about how the living organism works.

People have experimented with freezing a section of an object—say, the liver of a planarian, a lobster neuron, the eye of the *Daphnia*—taking extremely thin slices of it, dyeing them, and then physically hand-tracing all the required tissue until a model is built up. The problem with slices is that all these damn things are really three-dimensional; there are interconnections between the slices. And what you're trying to get is a three-dimensional structure of how the nerves connect. So you slice the thing the other way and physically hand-trace all of that.

The whole thing is a very long and involved process, and, unfortunately, even the best computers aren't good enough to do it automatically. You need human beings to do the tracing onto large digitizing tablets, and after that the spaces in between have to be filled in. Then, and only then, can you get a three-dimensional rotating model of the eye of the *Daphnia*.

Now what I am suggesting is that if you do get all this material in, you can create an electronic model on a chip that's not much bigger than the eye of the *Daphnia* itself, and you can check it out. It may be that you've missed a lot, but that will only prove that you've missed a lot. Or that the thing doesn't act the way you think it does. This isn't something that works or doesn't work.

I don't think tomorrow somebody's going to take a slice of Albert Einstein's brain and recreate Einstein by putting him back together again slice by slice and making an Einstein chip that will go on worrying about the general theory of relativity. But I do think that when you start doing this kind of reconstruction on any scale at all, if you can successfully recreate one nerve cell gap in any living organism, you've gone a long way towards taking a brain and recreating it.

True, the problems are infinite, but look where we've gone since the turn of the century, when we couldn't fly. Now we're testing our manned spacecraft for shuttling cargo around. As someone has pointed out, if we had gone into aerospace technology the way we went into computer technology, we would have been on the moon the day after Kitty Hawk. So I think we are going to see computers used in biological science not only to bring in and clarify and classify data, but to model the classified data as well. I also think we're going to see the building of specialized electronic devices that model biological systems so that the biological systems can be checked out.

For all we know, the eye of the *Daphnia* may be the most efficient way of building certain sensory equipment, and we just haven't found out about it yet. There's a lot to be said for what gets done in nature in very small packages. And if we're going to go on in miniaturization, that's one of the things we have to consider, especially since we've started to seriously consider parallel-processing devices, which mimic more of the brain structure than serial-processing devices have mimicked to date. It's something to be examined even more closely when we begin to consider matrix processing.

More intriguing, this replication would be a giant step towards the development of operative if not necessarily understandable cyborgian neural systems. Neurological prosthesis on a level complex enough for human system implantation is a considerably more esoteric development than, say, a heart pacer. Given the fact that heart pacers were beyond even conceptualization only a century ago, however, who is to say that chip-constructed electronic models of organic organisms would not at the very least lead to amazing new A to D peripherals within a decade? Beyond that, new mythological electronic beasts do not become unimaginable when you look at the future through the eye of a *Daphnia*. ▼

# PROMpuzzle

by Daniel Alber

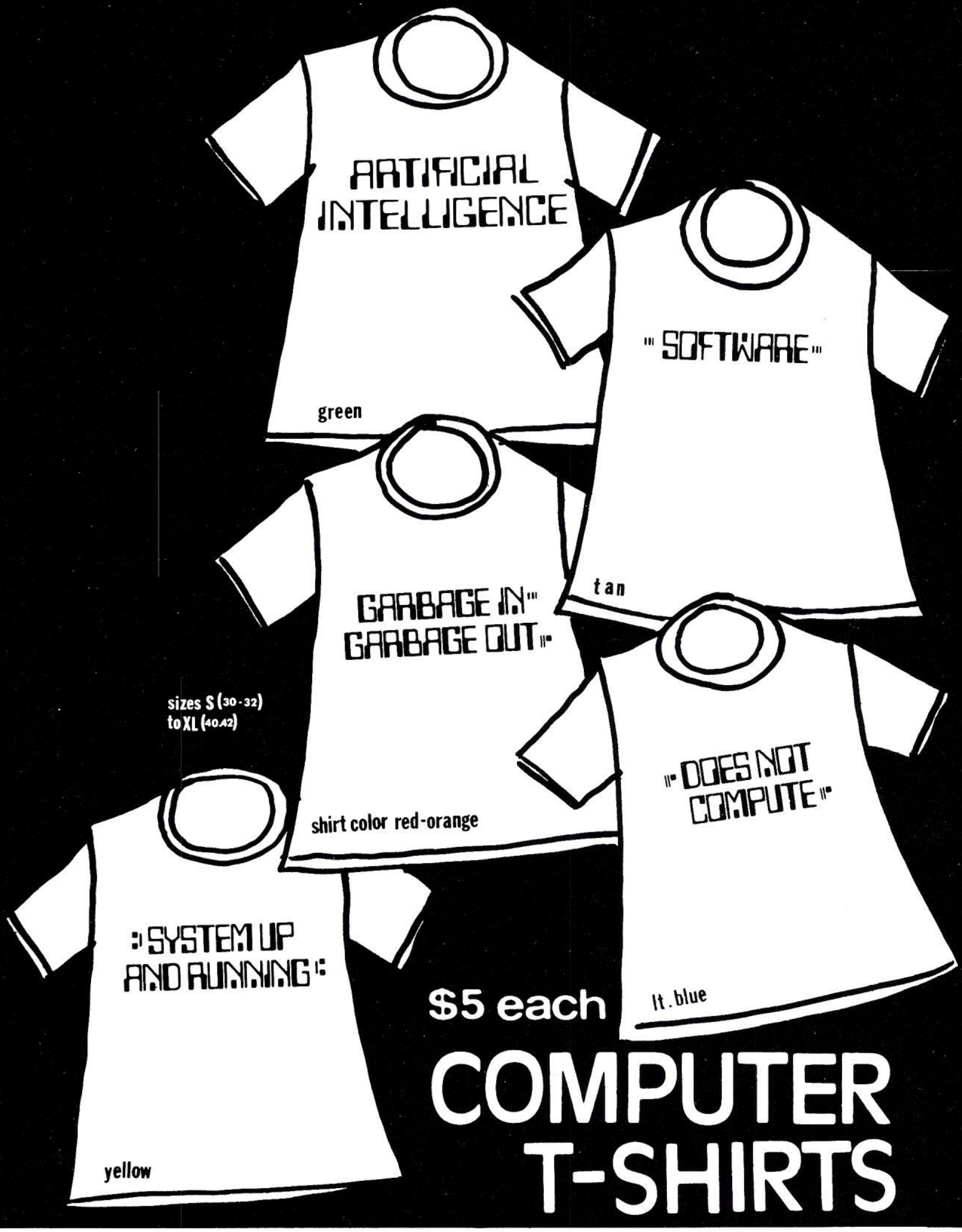
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | ■ | 5 | 6 | 7 | ■ | 8 | 9 | 10 | ■ | 11 | 12 | 13 | 14 |
| 15 |  |  |  | ■ | 16 |  |  | ■ | 17 |  |  | ■ | 18 |  |  |  |
| 19 |  |  |  | ■ | 20 |  |  | 21 |  |  |  | ■ | 22 |  |  |  |
| 23 |  |  |  | 24 | ■ | ■ | 25 |  |  | ■ | ■ | 26 |  |  |  |  |
| ■ | ■ | ■ | 27 |  | 28 | 29 |  |  | ■ | 30 | 31 |  |  | ■ | ■ | ■ |
| 32 | 33 | 34 | ■ | 35 |  |  | ■ | ■ | ■ | 36 |  |  | ■ | 37 | 38 | 39 |
| 40 |  |  | ■ | 41 |  |  | 42 | 43 | ■ | 44 |  |  | ■ | 45 |  |  |
| 46 |  |  | 47 |  | ■ | ■ | 48 |  | 49 |  |  | ■ | 50 |  |  |  |
| ■ | ■ | 51 |  |  | ■ | 52 |  |  |  |  | ■ | 53 |  |  | ■ | ■ |
| 54 | 55 |  |  | ■ | 56 |  |  |  |  | ■ | ■ | 57 |  |  | 58 | 59 |
| 60 |  |  | ■ | 61 |  |  | ■ | 62 |  | 63 | 64 |  | ■ | 65 |  |  |
| 66 |  |  | ■ | 67 |  |  | ■ | ■ | ■ | 68 |  |  | ■ | 69 |  |  |
| ■ | ■ | ■ | 70 |  |  |  | ■ | 71 | 72 |  |  |  | 73 | ■ | ■ | ■ |
| 74 | 75 | 76 |  |  | ■ | ■ | 77 |  |  | ■ | ■ | 78 |  | 79 | 80 | 81 |
| 82 |  |  |  | ■ | 83 | 84 |  |  |  | 85 | 86 | ■ | 87 |  |  |  |
| 88 |  |  |  | ■ | 89 |  |  | ■ | 90 |  |  | ■ | 91 |  |  |  |
| 92 |  |  |  | ■ | 93 |  |  | ■ | 94 |  |  | ■ | 95 |  |  |  |

ACROSS

1. Interpretive computer language
5. Degrees (abbr.)
8. Form of pulse modulation (abbr.)
11. Glut
15. Afloat
16. Made in____
17. Mature
18. Yesterday in Paris
19. Intellects
20. A non-metallic element having semiconducting properties
22. Mathematical chances
23. Solid ____ memory
25. Common suffix
26. Destroy programmed data
27. Read-only memories
30. Dunce
32. Hack
35. Label
36. Government agency (abbr.)
37. Linear quantizer
40. Baba or Muhammad
41. Tape updates
44. Bull ring cheer
45. Actor Wallach
46. ____ tape reader and monitor
48. Useful
50. Fixed attenuators
51. No longer in use (abbr.)
52. River bank
53. Biblical boat
54. Tulip seed
56. Copier powder
57. Approaches
60. ____ __ body kiss a body (2 wds., variant)
61. ____ Vegas
62. The march king
65. Sticky stuff
66. Craggy hill
67. Not downs
68. Simulation oriented language
69. Printer's measures
70. Victim
71. ____ access memory
74. ____ frequency distortion
77. Golden State (abbr.)
78. Circuits with one output and several inputs
82. Toga
83. Unbalanced electricity
87. ____ allocation
88. Actor Guinness
89. Broadcast
90. ____ pro nobis
91. Carry on
92. New York team
93. Remedy for short
94. Down time
95. Poems

DOWN

1. Lenz's and Laplace's
2. What ____? (2 wds.)
3. Bristle
4. Glue
5. Public transport
6. ____ was saying (2 wds.)
7. "Witch" city
8. Treaty
9. In the past
10. Guys
11. ____ circuit impedance
12. Verdi opera
13. Senator Kennedy et. al.
14. Gaelic
21. ____ and outs
24. Diners
26. Otherwise
28. Rapid access disc
29. Sash
30. Device to channel current flow
31. Paris airport
32. Condenser for short
33. ____ mode (2 wds.)
34. MOSFET alternative
37. Undesirable conductive paths
38. Ancient
39. Poetic contraction
42. Not now
43. Rescues
47. Decrease
49. Famed fiddler
50. Prefix meaning before
52. Computer energy dissipator
53. Physical system in electronic computers
54. Binary digit (abbr.)
55. Flying saucer
56. Storage device
58. Wonderful computer magazine
59. Distress signal
61. Entice
63. Branch of the military (abbr.)
64. Turf
70. Picoseconds for short
71. ROM address register
72. An international program language
73. Type of coding
74. Carriage
75. Punch card feature
76. Help
77. Machine-processable storage medium
79. Frog
80. Sea bird
81. Groups of components
83. Retrieval device
84. Rush
85. Epoch
86. Droop


ARTIFICIAL INTELLIGENCE
green
" SOFTWARE "
tan
" GARBAGE IN "
GARBAGE OUT "
shirt color red-orange
" DOES NOT COMPUTE "
lt. blue
: SYSTEM UP AND RUNNING :
yellow

sizes S (30-32)
to XL (40-42)

$5 each

# COMPUTER T-SHIRTS

name ______________________

address ______________________

city __________ state ____ zip ______

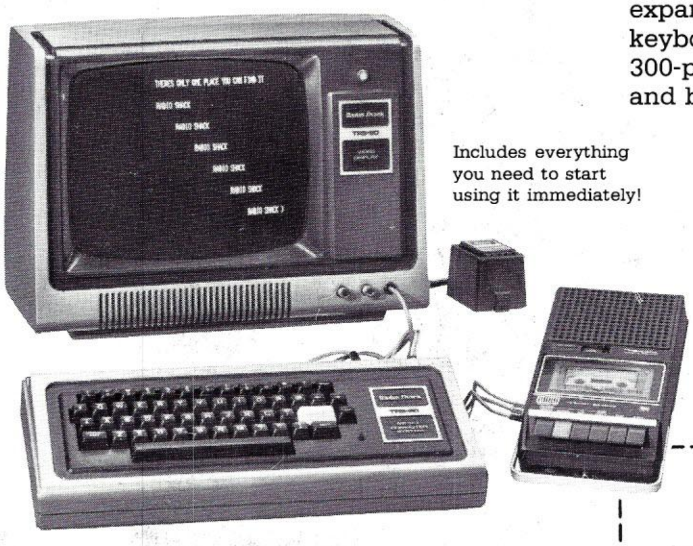
mail to MARTHA HERMAN 114 W. 17 STREET
NEW YORK 10011
(212) 691-2821

| quantity | chest size | design | $ | |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| add .60 per shirt for shipping | | | | |
| total enclosed | | | | |

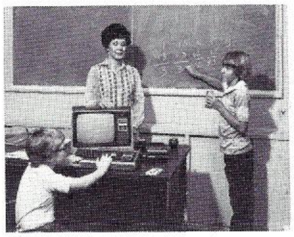
# The first complete low-cost microcomputer system for home, business or education!

## Radio Shack TRS-80

The TRS-80 is for people who want to use a computer now — without the delay, work and problems of building one. The system is fully wired, tested and U.L. listed — ready for you to plug in and use! Program it to handle your personal finances, small business accounting, teaching functions, kitchen computations, innumerable games — and use Radio Shack's expanding line of prepared programs on cassettes. The Z80-based system comes with 4K read/write memory and Radio Shack Level-I BASIC stored in read-only memory. Memory expandable to 62K bytes. Includes CPU, memory, keyboard, display, power supply, cassette data recorder, 300-page manual, 2-game cassette program. Designed and built in USA by Radio Shack. Only 599.95.



Includes everything you need to start using it immediately!



*Clip and Mail Coupon Today!*

C012

Mail to: Radio Shack, Dept. TRS-80
205 N.W. 7th St., Ft. Worth, TX 76101

**Send me more data on the TRS-80 microcomputer**

- **Description of applications, software and peripherals available through Radio Shack • Owners' newsletter**
- **Price list • List of stocking stores and dealers**

NAME ______________________ APT. NO. ______
ADDRESS ______________________
CITY ______________ STATE ________ ZIP ________

SOLD ONLY WHERE YOU SEE THIS SIGN:

**Radio Shack®**

A TANDY COMPANY • FORT WORTH, TEXAS 76102
OVER 6000 LOCATIONS IN NINE COUNTRIES

Price may vary at individual stores and dealers